Você pode utilizar formulários para oferecer a seus usuários uma interface para visualizar e inserir informações em bancos de dados. Entretanto, os formulários podem ser muito mais do que uma simples interface: oferecem uma rica variedade de objetos capazes de responder aos eventos do usuário (ou do sistema), de modo a facilitar as tarefas de gerenciamento de informações.

Este capítulo abrange os seguintes tópicos:

Estruturando formulários

O Visual FoxPro possui um poderoso **Criador de formulários** que torna a criação de formulários uma tarefa rápida e fácil. Você pode ter:

- Vários tipos de objetos nos formulários.
- Dados ligados a objetos no formulário.
- Formulários filho ou de nível máximo.
- Formulários múltiplos que podem ser manipulados em conjunto.
- Formulários baseados nos seus próprios modelos personalizados.

Formulários e conjuntos de formulários são objetos cujas propriedades, eventos e métodos podem ser definidos no **Criador de formulários**. Os conjuntos de formulários são formados por um ou mais formulários que podem ser manipulados como uma unidade. Por exemplo, se você tiver um conjunto composto por quatro formulários, poderá exibi-los ou ocultá-los como uma unidade com apenas um comando, em tempo de execução.

Criando formulários novos

Você pode criar novos formulários no **Criador de formulários** e ver como cada objeto aparecerá para o usuário à medida que você o cria.

Para criar formulários novos

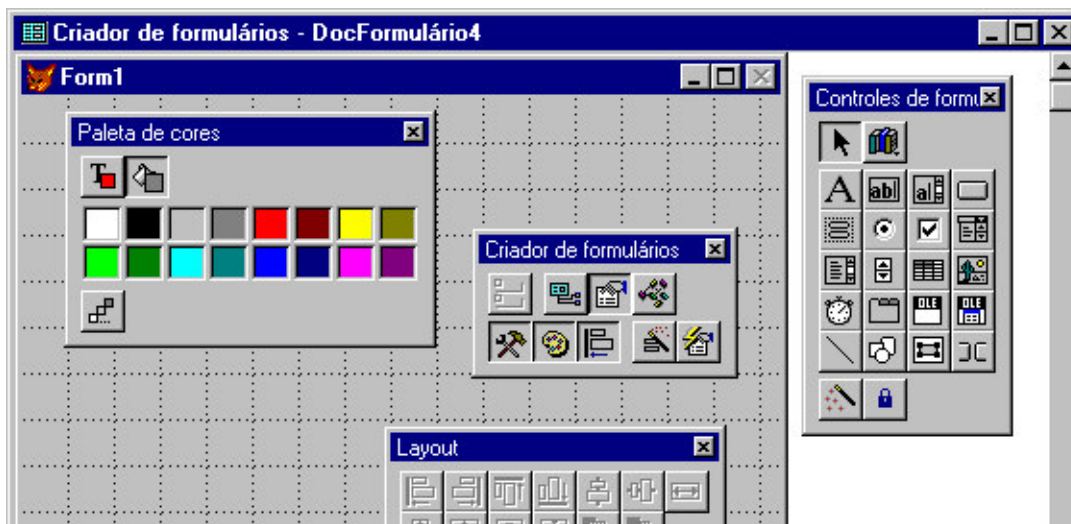
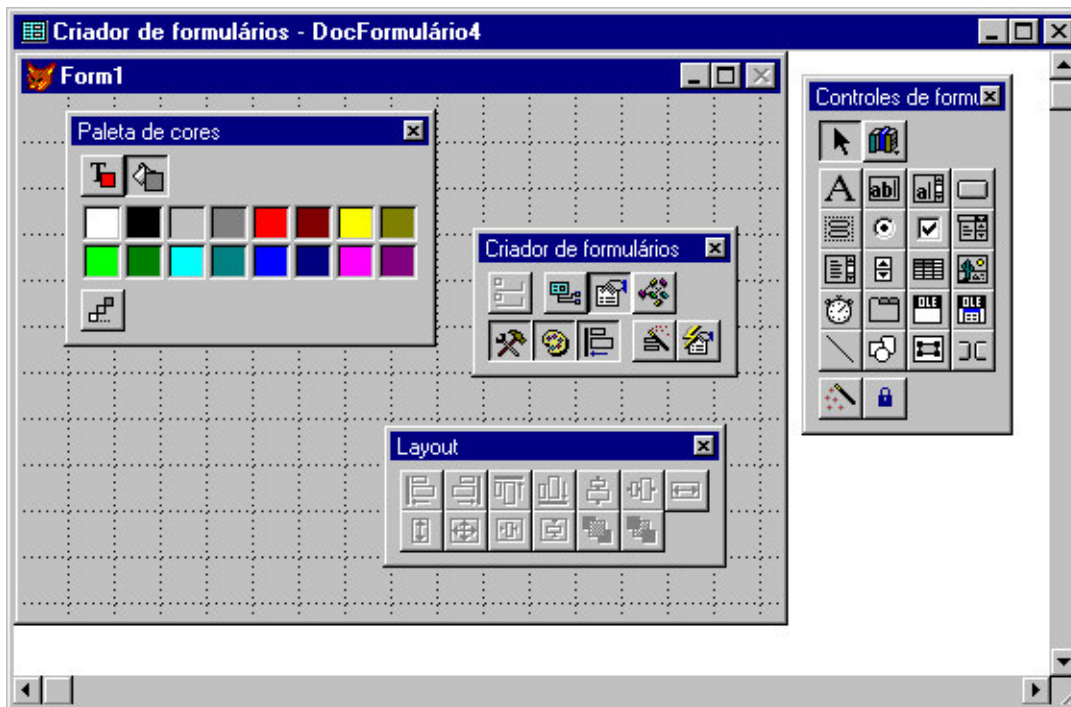
- No **Gerenciador de projetos**, selecione a opção **Formulários** e escolha **Novo**.

  – Ou –

- No menu **Arquivo**, escolha **Novo**, selecione **Formulário** e, em seguida, escolha **Novo arquivo**.

  – Ou –

- Utilize o comando CREATE FORM.

Criador de formulários, com suas Barras de ferramentas: criador de formulários, controles, layout e paleta de cores

Para obter uma descrição mais detalhada sobre o **Criador de formulários**, consulte o capítulo 8, Gerenciando dados através de formulários, no *Guia do Usuário*. Para obter maiores informações sobre as barras de ferramentas, procure "barras de ferramentas" na Ajuda e selecione a barra de ferramentas sobre a qual precisa de informações.

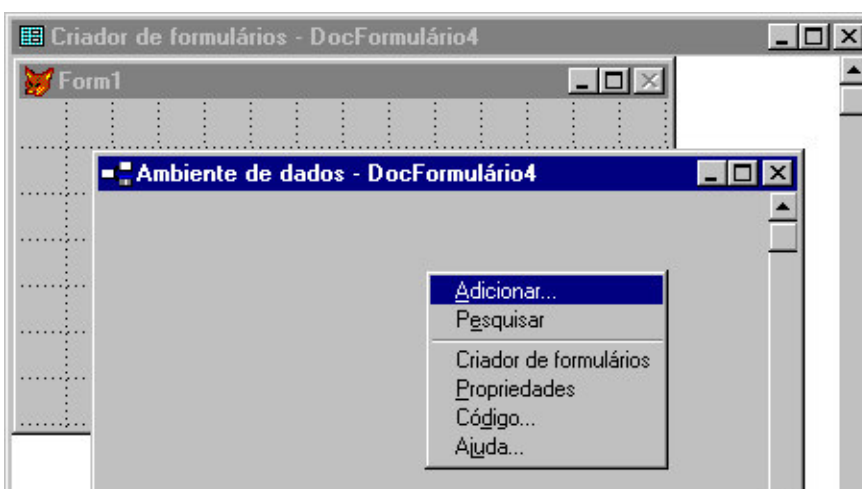
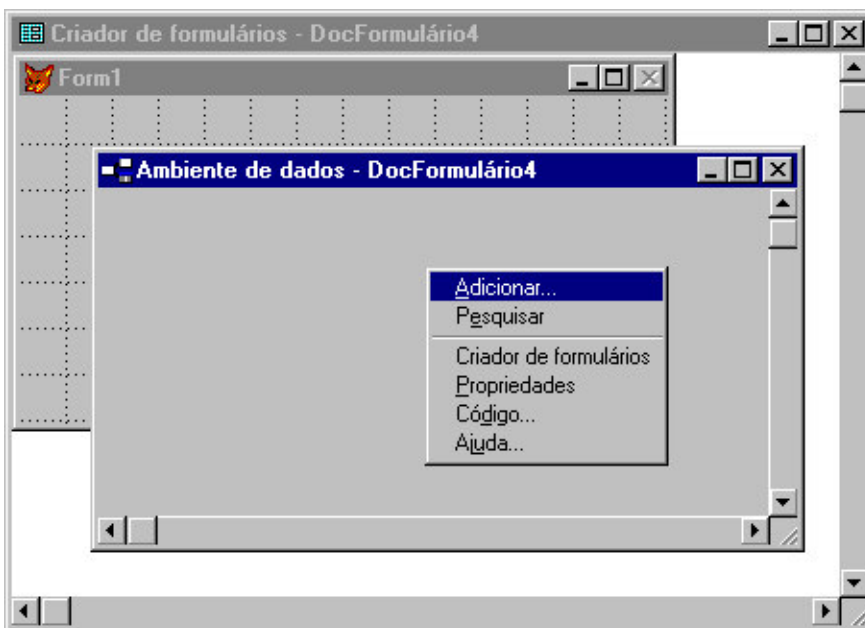
Definindo o Ambiente de dados

Cada formulário ou conjunto de formulários inclui um ***Ambiente de dados***. O **Ambiente de dados** é um objeto que inclui as tabelas ou visualizações ,com as quais o formulário interage, e os relacionamentos entre as tabelas esperados pelo formulário. Você pode preparar visualmente o **Ambiente de dados** no **Criador de ambientes de dados** e salvá-lo com o formulário.

O **Ambiente de dados** pode automatizar a abertura e o fechamento de tabelas e visualizações quando o formulário estiver sendo executado. Além disso, o **Ambiente de dados** irá ajudá-lo a definir a propriedade ControlSource para controles, preenchendo a caixa da definição da propriedade ControlSource na janela **Propriedades** com todos os campos do **Ambiente de dados**.

Para abrir o Criador de ambientes de dados

1 No menu **Exibir**, selecione a opção **Ambiente de dados**.
2 No menu **Tecla de atalho**, escolha **Adicionar**.
3 Na caixa de diálogo **Abrir**, escolha uma tabela ou visualização a ser adicionada ao **Ambiente de dados**.

Criador de ambientes de dados

Propriedades comuns de ambientes de dados

As propriedades de ambientes de dados a seguir são normalmente definidas na janela **Propriedades**:

Propriedade	Descrição	Definição padrão
AutoCloseTables	Define se as tabelas e visualizações deverão ser fechadas ou não quando o formulário for liberado.	Verdadeiro (.T.)
AutoOpenTables	Define se as tabelas e visualizações do ambiente de dados deverão ser abertas quando o formulário for executado.	Verdadeiro (.T.)
InitialSelectedAlias	Especifica a tabela ou visualização que é selecionada ao ser executado o formulário.	"" na criação. Se não for especificado, a seleção inicial em tempo de execução será o primeiro Cursor adicionado ao DataEnvironment.

Adicionando uma tabela ou visualização ao Criador de ambientes de dados

Ao adicionar tabelas ou visualizações ao **Criador de ambientes de dados**, você pode ver os campos e os índices pertencentes à tabela ou visualização.

Para adicionar uma tabela ou visualização ao ambiente de dados

1 No **Criador de ambientes de dados**, selecione a opção **Adicionar** no menu **AmbienteDados**.

2 Na caixa de diálogo **Adicionar tabela ou visualização**, escolha uma tabela ou visualização na lista.

– Ou –

Se não houver um banco de dados ou projeto aberto, selecione a opção **Outros** para escolher

uma tabela.

Você também pode arrastar uma tabela ou visualização de um projeto que estiver aberto para o **Criador de banco de dados** no **Criador de ambientes de dados**.

Se o **Criador de ambientes de dados** estiver ativo, a janela **Propriedades** exibirá objetos e propriedades associados ao **Ambiente de dados**. Cada tabela ou visualização do **Ambiente de dados**, cada relacionamento entre tabelas e o **Ambiente de dados** propriamente dito é um objeto independente na caixa **Objeto** da janela **Propriedades**.

Removendo uma tabela do Criador de ambientes de dados

Ao remover uma tabela do **Ambiente de dados**, todos os relacionamentos relativos a essa tabela também serão removidos.

Para remover tabelas ou visualizações do Criador de ambientes de dados

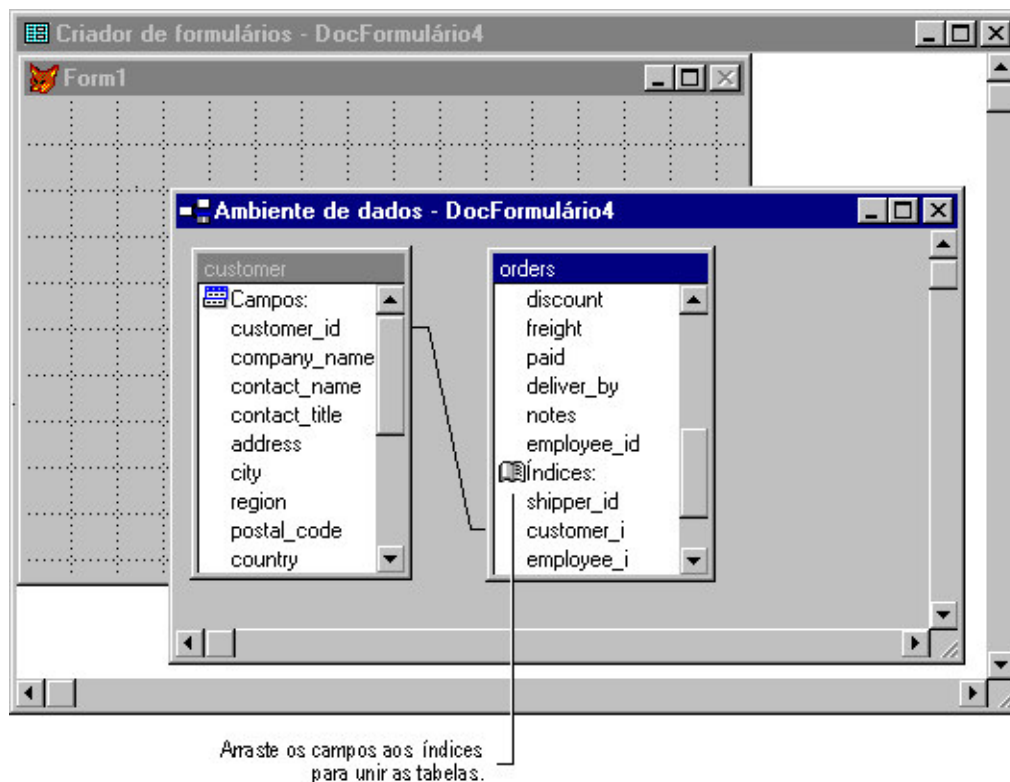
1. No **Criador de ambientes de dados**, selecione a tabela ou visualização.
2. No menu **AmbienteDados**, selecione a opção **Remover**.

Definindo relacionamentos no Criador de ambientes de dados

Se você adicionar ao **Criador de ambientes de dados** tabelas com relacionamentos permanentes definidos em um banco de dados, esses relacionamentos serão adicionados automaticamente ao **Ambiente de dados**. Mesmo se as tabelas não tiverem relacionamentos permanentes, você poderá relacioná-las no **Criador de ambientes de dados**.

Para definir relacionamentos no Criador de ambientes de dados

- Arraste um campo da tabela primária até a marca de índice correspondente na tabela relacionada.

Criador de ambientes de dados com relacionamentos definidos entre tabelas

Também é possível arrastar um campo da tabela primária até um campo na tabela relacionada. Se

não houver uma marca de índice na tabela relacionada que corresponda ao campo da tabela primária, você será solicitado a criar a marca de índice.

Editando relacionamentos no Criador de ambientes de dados

Quando você definir uma relação no **Criador de ambientes de dados**, surgirá uma linha entre as tabelas, indicando esse relacionamento.

Para editar as propriedades da relação

- Na janela **Propriedades**, selecione a relação na caixa **Objeto**.

As propriedades da relação correspondem a cláusulas e palavras-chave nos comandos SET RELATION e SET SKIP.

A propriedade RelationalExpr é definida como padrão como o nome do campo de chave primária da tabela primária. Se a tabela relacionada estiver indexada em uma expressão, você deverá definir a propriedade RelationalExpr como essa expressão. Por exemplo, se a tabela relacionada estiver indexada em `UPPER(cust_id)`, você precisará definir RelationalExpr como `UPPER(cust_id)`.

Se a relação não for um relacionamento um-para-n, defina a propriedade OneToMany como Falso (.F.). Isso corresponde à utilização do comando SET RELATION sem utilizar o comando SET SKIP.

Definir a propriedade OneToMany de uma relação como verdadeiro (.T.) corresponde a executar o comando SET SKIP. Quando você passa pela tabela pai, o ponteiro do registro permanece no mesmo registro pai, até que o ponteiro do registro passe por todos os registros relacionados da tabela filho.

Observação Caso deseje um relacionamento um-para-n no formulário, defina a propriedade OneToMany como verdadeiro (.T.), mesmo que um relacionamento um-para-n permanente tenha sido estabelecido em um banco de dados.

Criando interfaces de um e de vários documentos

O Visual FoxPro permite que você crie dois tipos de aplicativos:

- Os aplicativos de *interface de múltiplos documentos(MDI)* são compostos por uma janela principal e as janelas do aplicativo estão contidas, ou flutuando, na parte superior da janela principal. O Visual FoxPro é principalmente um aplicativo MDI, com a janela de comando, janelas de edição e janelas de criadores contidas na janela principal do Visual FoxPro.
- Os aplicativos de *interface de documento único(SDI)* são compostos por uma ou mais janelas independentes, cada qual aparecendo separadamente na área de trabalho do Windows. O Microsoft Exchange é um exemplo de um aplicativo SDI, no qual cada mensagem aberta é exibida em sua própria janela independente.

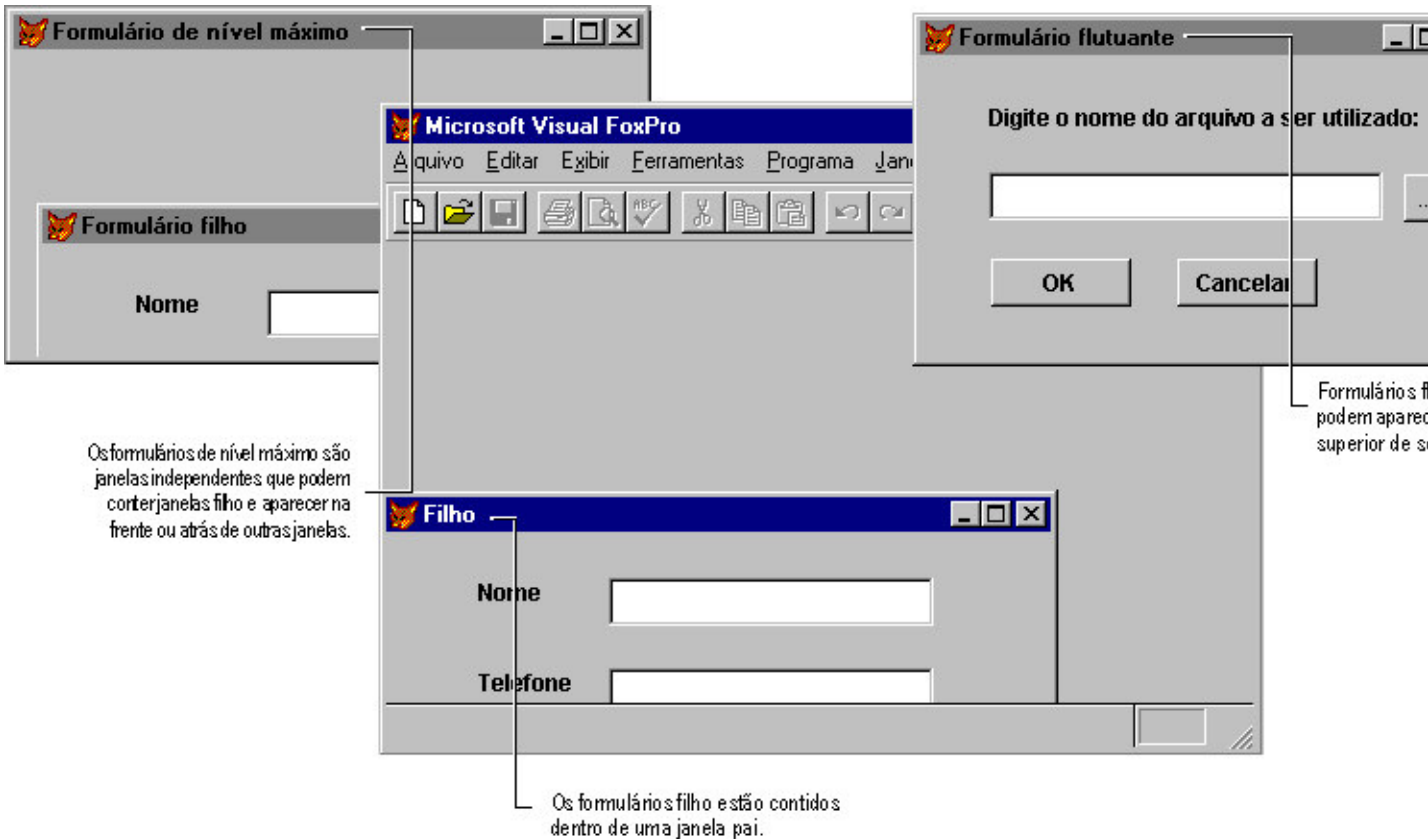
Um aplicativo composto por uma única janela geralmente é do tipo SDI, mas alguns aplicativos misturam os elementos de SDI e MDI. Por exemplo, o Visual FoxPro exibe seu depurador como um aplicativo SDI, que por sua vez contém janelas MDI de sua propriedade.

Para suportar os dois tipos de interface, o Visual FoxPro permite que você crie vários tipos de formulários:

- *Formulário filho*. Um formulário contido em outra janela, utilizado na criação de aplicativos MDI. Os formulários filho não podem ser movidos para fora dos limites do formulário pai (o formulário principal) e quando minimizados aparecem na parte inferior do formulário pai. Se o formulário pai for minimizado, eles também serão.
- *Formulário flutuante*. Um formulário que pertence a um formulário pai (principal), mas não está contido em seu interior. Em vez disso, os formulários flutuantes podem ser movidos para qualquer local na tela, porém não podem ser movidos para trás da sua janela pai. Se minimizado, um formulário flutuante aparece na parte inferior da área de trabalho. Se o formulário pai for minimizado, os formulários flutuantes também serão. Além disso, os formulários flutuantes são

utilizados na criação de aplicativos MDI.

- *Formulário de nível máximo*. Um formulário independente - sem formulário pai - utilizado para criar um aplicativo SDI ou para servir como o pai de outros formulários filho em um aplicativo MDI. Os formulários de nível máximo funcionam no mesmo nível de outros aplicativos do Windows e podem aparecer na frente ou atrás deles. São exibidos na barra de tarefas do Windows.

Formulários filho, flutuantes e de nível máximo

Especificando um tipo de formulário

Todos os tipos de formulários são criados quase da mesma forma, mas são definidas propriedades especificas para indicar como o formulário deve se comportar.

Ao criar um formulário filho, especifique não só que ele deve aparecer dentro de outro formulário, como também que se trata de um formulário filho compatível com MDI, que indica como o formulário se comporta quando minimizado. Se o formulário filho for compatível com MDI, combinará com o formulário pai, compartilhando a barra de título, legenda, menus e barras de ferramentas do formulário pai. Um formulário filho não compatível com MDI é maximizado na área cliente completa do formulário pai, mas mantém a sua própria legenda e barra de título.

Para especificar um formulário filho

1 Crie ou edite o formulário utilizando o **Criador de formulários**.

2 Defina a propriedade ShowWindow do formulário como um dos valores a seguir:

- **0 – Na tela**. O pai do formulário filho será a janela principal do Visual FoxPro.
- **1 – Em Formulário de nível máximo.** O pai do formulário filho será o formulário de nível máximo ativo quando a janela filho estiver exibida. Utilize esta definição se desejar que a janela filho apareça em qualquer janela de nível máximo em vez de aparecer na janela principal do Visual FoxPro.

3 Defina a propriedade MDIForm do formulário como .T. (verdadeiro) se desejar que o formulário filho seja combinado com o pai quando maximizado, ou como .F. (falso) se a janela filho precisar ser mantida como uma janela separada quando maximizada.

Um formulário flutuante é uma variação de um formulário filho.

Para especificar um formulário flutuante

1 Crie ou edite o formulário utilizando o **Criador de formulários**.

2 Defina a propriedade ShowWindow do formulário como um dos valores a seguir:

- **0 – Na Tela**. O pai do formulário flutuante será a janela principal do Visual FoxPro.
- **1 – Em Formulário de nível máximo**. O pai do formulário flutuante será o formulário de nível máximo ativo quando a janela flutuante estiver exibida.

3 Defina a propriedade Desktop do formulário como .T. (verdadeiro).

Para especificar um formulário de nível máximo

1 Crie ou edite o formulário utilizando o **Criador de formulários**.

2 Defina a propriedade ShowWindow do formulário como **2 – Como Formulário de nível máximo.**

Exibindo um formulário filho dentro de um formulário de nível máximo

Se você criou um formulário filho cuja propriedade ShowWindow está definida como **1 – Em Formulário de nível máximo**, você não especificará diretamente o formulário de nível máximo que age como o pai do formulário filho. Em vez disso, o Visual FoxPro atribui o formulário filho a um pai no momento em que a janela filho estiver exibida.

Para exibir um formulário filho dentro de um formulário de nível máximo

1 Crie um formulário de nível máximo.

2 No código de evento do formulário de nível máximo, inclua o comando DO FORM, especificando o nome do formulário filho a ser exibido.

Por exemplo, crie um botão no formulário de nível máximo e, em seguida, no código de evento Click do botão, inclua um comando como este:

```
DO FORM MyChild
```

Observação O formulário de nível máximo deve estar visível e ativo quando o formulário filho for exibido. Logo, você não poderá utilizar o evento Init do formulário de nível máximo para exibir um formulário filho, pois o formulário de nível máximo ainda não estará ativo.

3 Ative o formulário de nível máximo e, em seguida, se necessário, dispare o evento que exibe o formulário filho.

Ocultando a janela principal do Visual FoxPro

Se você estiver executando um formulário de nível máximo, é possível que você não deseja que a janela principal do Visual FoxPro fique visível. Você pode utilizar a propriedade Visible do objeto Application para ocultar e exibir a janela principal do Visual FoxPro, conforme necessário.

Para ocultar a janela principal do Visual FoxPro

1 No evento Init do formulário, inclua a seguinte linha de código:

```
Application.Visible = .F.
```

2 No evento Destroy do formulário, inclua a seguinte linha de código:

```
Application.Visible = .T.
```

Certifique-se de que também tenha fornecido um meio de fechar o formulário utilizando `THISFORM.Release` em algum método ou evento.

Observação Você também pode incluir a seguinte linha em um arquivo de configuração para ocultar a janela principal do Visual FoxPro:

```
SCREEN = OFF
```

Para obter maiores informações sobre como configurar o Visual FoxPro, consulte o capítulo 3, Configurando o Visual FoxPro, no *Guia de Instalação e Índice Mestre*.

Adicionando um menu a um formulário de nível máximo

Para adicionar um menu a um formulário de nível máximo

1. Crie um menu de formulário de nível máximo. Para obter maiores informações sobre como criar menus para formulários de nível máximo, consulte o capítulo 11, Criando menus e barras de ferramentas
2. Defina a propriedade ShowWindow do formulário como **2 – Como Formulário de nível máximo**.
3. No evento Init do formulário, execute o programa de menus e dê dois parâmetros a ele:

   DO *menuname.mpr* WITH *oForm, lAutoRename*
   oForm é a referência de um objeto no formulário. No evento Init do formulário, passe THIS como o primeiro parâmetro.

   lAutoRename especifica se um novo nome exclusivo será gerado ou não para o menu. Se você planeja executar várias instâncias do formulário, passe .T. para *lAutoRename.*

Por exemplo, você pode chamar um menu denominado `mySDImenu` com esse código:

```
DO mySDImenu.mpr WITH THIS, .T.
```

Estendendo formulários com conjuntos de formulários

Você pode manipular vários formulários como se fossem um só, incluindo-os em um conjunto de formulários. Os conjuntos de formulários apresentam as seguintes vantagens:

- Pode-se exibir ou ocultar todos os formulários de um conjunto de uma só vez.
- Pode-se organizar visualmente os formulários múltiplos, de uma só vez, para controlar suas posições relativas.
- Como todos os formulários em um conjunto de formulários são definidos em um único arquivo .SCX com um único ambiente de dados, é possível sincronizar automaticamente os ponteiros do registro de vários formulários. Se for alterado o ponteiro de registro na tabela pai de um formulário, os registros filho de outro formulário também serão atualizados e exibidos.

Observação Ao executar um conjunto de formulários, todos os formulários e objetos a ele pertencentes serão carregados. O carregamento de muitos formulários com vários controles poderá demorar alguns segundos.

Criando um novo conjunto de formulários

Um conjunto de formulários é um recipiente pai de um ou mais formulários. Estando no **Criador de formulários**, você pode criar conjuntos de formulários.

Para criar um conjunto de formulários

- No menu **Formulário**, selecione a opção **Criar conjunto de formulários**.

Se você não desejar trabalhar com vários formulários em um grupo de formulários, não precisará criar um conjunto de formulários. Uma vez que você tenha criado um conjunto, poderá adicionar novos formulários ao mesmo.

Adicionando e removendo formulários

Tendo criado um conjunto de formulários, você pode adicionar e remover formulários.

Para adicionar outros formulários ao conjunto

- No menu **Formulário**, selecione a opção **Adicionar novo formulário**.

Para remover um formulário do conjunto de formulários

1 Na caixa **Formulário** na parte inferior do **Criador de formulários**, selecione o formulário.

2 No menu **Formulário**, selecione a opção **Remover formulário**.

Se houver apenas um formulário no conjunto, você poderá remover o conjunto de formulários, para ficar apenas com o formulário.

Para remover um conjunto de formulários

- No menu **Formulário**, selecione a opção **Remover conjunto de formulários**.

Os formulários são salvos no formato de tabela em um arquivo com a extensão .SCX. Ao ser criado um formulário, a tabela .SCX contém um registro do formulário, um registro do ambiente de dados e dois registros reservados para uso interno. Um registro é adicionado para cada objeto adicionado ao formulário ou ao ambiente de dados. Se você criar um conjunto de formulários, será adicionado um registro adicional para o conjunto de formulários e para cada um dos novos formulários. O recipiente pai de cada formulário é o conjunto de formulários. O recipiente pai de cada controle é o formulário no qual é colocado.

Dica Ao executar um conjunto de formulários, talvez você não deseje que todos os formulários do conjunto fiquem imediatamente visíveis. Defina a propriedade Visible como Falso (.F.) para os formulários que você não desejar que sejam exibidos ao ser executado o conjunto. Defina a propriedade Visible como verdadeiro, (.T.) quando desejar que os formulários sejam exibidos.

Adicionando objetos a formulários

Para criar a funcionalidade desejada no formulário, você adiciona os controles adequados, define as propriedades do formulário e dos controles e escreve o código de evento.

Pode-se inserir os seguintes tipos de objetos nos formulários:

- Controles
- Recipientes
- Classes definidas pelo usuário
- Objetos OLE

Conhecendo objetos recipientes e de controle

No Visual FoxPro, os objetos pertencem a uma de duas categorias, dependendo da natureza da classe na qual se baseiam:

- Os *recipientes* podem armazenar outros recipientes ou controles. Podem ser o objeto pai de outros objetos. Por exemplo, um formulário, como recipiente, é o objeto pai de uma caixa de verificação desse formulário.
- Os *controles* podem estar contidos em recipientes, mas não podem ser pais de outros objetos. Por exemplo, uma caixa de verificação não pode conter qualquer outro objeto.

O **Criador de formulários** permite a criação de recipientes e controles.

Recipiente	Pode conter
Coluna	Cabeçalhos e qualquer objeto, à exceção de conjuntos de formulários, formulários, barras de ferramentas, cronômetros e outras colunas
Grupo de botões de comando	Botões de comando
Conjunto de formulários	Formulários, barras de ferramentas
Formulário	Molduras de página, grades, qualquer controle
Grade	Colunas

Grupo de botões de opção	Botões de opção
Moldura de página	Páginas
Página	Grades, qualquer controle

Adicionando recipientes do Visual FoxPro

Além de formulários e conjuntos deles, o Visual FoxPro oferece quatro classes recipientes principais.

Classes recipientes do Visual FoxPro

Command button group	Option button group
Grid	Page frame

Para adicionar objetos recipientes a um formulário

- Na **Barra de ferramentas controles de formulário**, selecione o botão de objeto recipiente desejado (grade, moldura de página ou grupo de botões) e arraste para dimensioná-lo no formulário.

Quando você adicionar um grupo de botões de comando ou um grupo de botões de opção a um formulário no **Criador de formulários**, esse grupo conterá, como padrão, dois botões. Quando você adicionar uma moldura de página a um formulário, essa moldura conterá duas páginas, também como padrão. É possível adicionar mais botões ou páginas através da definição das propriedades ButtonCount ou PageCount como o número desejado.

Quando você adicionar uma grade a um formulário, a propriedade ColumnCount será definida, por padrão, como –1, o que indica AutoFill. Em tempo de execução, a grade exibirá um número de colunas igual ao número de campos existentes na tabela RowSource. Se você não desejar AutoFill, poderá especificar o número de colunas, definindo a propriedade ColumnCount da grade.

Para obter maiores informações sobre esses objetos recipientes, consulte o capítulo 10, Utilizando controles.

Propriedades de conjunto e de contagem

Todos os objetos recipientes do Visual FoxPro possuem uma propriedade de contagem e uma propriedade de conjunto a eles associadas. A propriedade de conjunto é uma matriz que faz referência a cada objeto contido. A propriedade de contagem é uma propriedade numérica que indica o número de objetos contidos.

As propriedades de conjunto e de contagem de cada recipiente são nomeadas de acordo com o tipo de objeto que pode ser contido no recipiente. A tabela a seguir lista os recipientes e as propriedades de conjunto e contagem correspondentes.

Recipiente	Propriedade de conjunto	Propriedade de contagem
Aplicativo	Objects Forms	Contagem FormCount
Conjunto de formulários	Forms	FormCount
Formulário	Objects Controls	Contagem ControlCount
Moldura de página	Pages	PageCount
Página	Controls	ControlCount
Grade	Columns	ColumnCount
Grupo de comandos	Buttons	ButtonCount
Grupo de opções	Buttons	ButtonCount
Coluna	Controls	ControlCount

Barra de ferramentas	Controls	ControlCount
Recipiente	Controls	ControlCount
Controle	Controls	ControlCount

Essas propriedades permitem que você utilize um loop para manipular de forma programática todos ou objetos recipientes específicos. Por exemplo, as linhas de código a seguir definem a propriedade BackColor de colunas em uma grade como a alternância entre verde e vermelho:

```
o = THISFORM.grd1
FOR i = 1 to o.ColumnCount
   IF i % 2 = 0 && Coluna com números pares
      o.Columns(i).BackColor = RGB(0,255,0) && Verde
   ELSE
      o.Columns(i).BackColor = RGB(255,0,0) && Vermelho
   ENDIF
ENDFOR
```

Adicionando controles do Visual FoxPro a um formulário

A **Barra de ferramentas controles** permite que você adicione facilmente qualquer um dos controles padrão do Visual FoxPro ao seu formulário.

Controles padrão do Visual FoxPro

Caixa de verificação	Imagem	Controle de Ligação de OLE	Caixa de texto
Caixa de combinação	Rótulo	Controle de Recipiente de OLE	Cronômetro
Botão de comando	Linha	Forma	
Caixa de edição	Caixa de listagem	Controle de rotação	

Para adicionar controles a um formulário

- Na **Barra de ferramentas controles de formulário**, selecione o botão de controle desejado e clique ou arraste para dimensioná-lo no formulário.

Para obter maiores informações sobre a escolha do controle, consulte o capítulo 10, Utilizando controles..

Adicionando controles acoplados a dados a um formulário

Você pode acoplar controles a dados em uma tabela, visualização, campo de tabela ou campo de visualização, definindo a propriedade ControlSource de um controle como um campo ou a propriedade RecordSource de uma grade como uma tabela ou visualização. Além disso, você pode criar controles acoplados a dados, arrastando campos ou tabelas para o formulário diretamente dos seguintes locais:

- **Gerenciador de projetos**
- **Criador de bancos de dados**
- **Criador de ambientes de dados**

A classe de controle criada desta forma depende das definições de Mapeamento de Campos na guia **Propriedades do criador de tabelas** ou na guia **Mapeamento de campos** da caixa de diálogo **Opções**.

Para obter maiores informações sobre como definir classes de controle padrão, consulte o **Criador de tabelas** ou a guia **Mapeamento de campos** da caixa de diálogo **Opções**.

Adicionando objetos definidos pelo usuário a um formulário

Um dos recursos mais poderosos do Visual FoxPro é a capacidade de criar classes que podem ser facilmente utilizadas e reutilizadas em diversas partes dos seus aplicativos. Uma vez criadas essas classes, você pode adicioná-las aos seus formulários.

Para adicionar um objeto baseado em uma classe personalizada

- No **Gerenciador de projetos**, arraste a classe até o recipiente.

Você poderá também adicionar suas classes diretamente da **Barra de ferramentas controles de formulário** quando adicioná-las à barra de ferramentas.

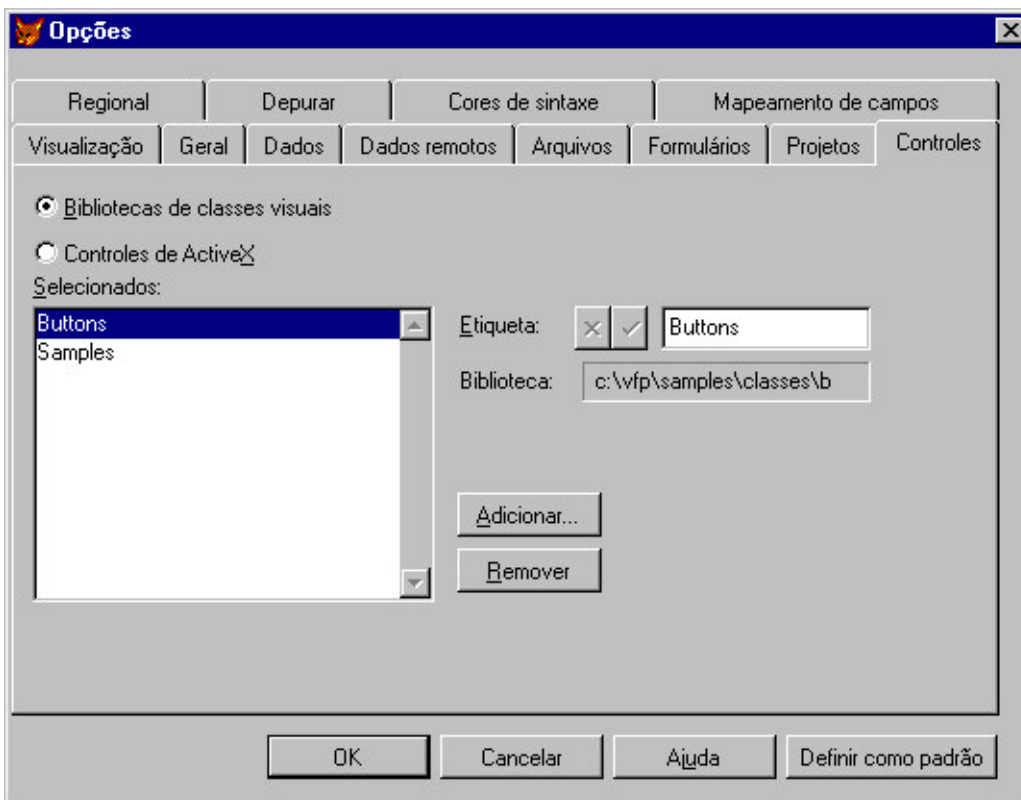
Adicionando bibliotecas de classes à Barra de ferramentas controles

Você precisa registrar suas bibliotecas de classes para que possam ser exibidas na **Barra de ferramentas controles de formulário**.

Para registrar uma biblioteca de classes

1. No menu **Ferramentas**, escolha **Opções**.
2. Na caixa de diálogo **Opções**, escolha a guia **Controles**.
3. Escolha **Adicionar**.
4. Na caixa de diálogo **Abrir**, escolha a biblioteca de classes a ser inserida na lista **Selecionados** e escolha **Abrir**.
5. Repita as etapas 3 e 4 até que todas as bibliotecas que desejar registrar tenham sido adicionadas.

As classes das bibliotecas de classes da lista Selecionados podem ser utilizadas no **Criador de formulários** com a mesma facilidade de utilização das classes principais do Visual FoxPro.

Guia Controles da caixa de diálogo Opções

Dica Se você desejar que as bibliotecas de classes fiquem disponíveis na **Barra de ferramentas controles de formulário** sempre que o Visual FoxPro for executado, escolha Definir Como Padrão

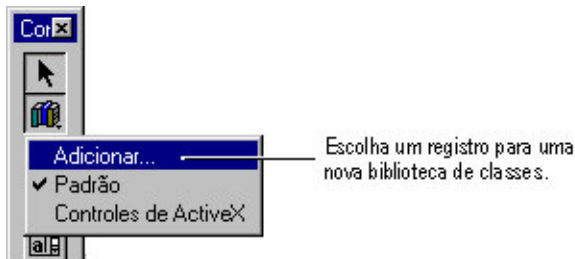
na caixa de diálogo **Opções**.

Você pode também registrar bibliotecas diretamente no **Criador de formulários**.

Para registrar uma biblioteca de classes no Criador de formulários

1 Na **Barra de ferramentas** **controles de formulário**, escolha o botão **Exibir classes**.

2 No submenu, escolha **Adicionar**.

Submenu do botão Exibir classes

3 Na caixa de diálogo **Abrir**, escolha uma biblioteca de classes a ser adicionada à **Barra de ferramentas controles de formulário** e selecione a opção **Abrir**.

Adicionando objetos a um formulário a partir de uma biblioteca de classes

Já tendo adicionado as bibliotecas de classes na guia **Classes** da caixa de diálogo **Opções** ou a partir do submenu **Exibir classes**, você poderá acessá-las no **Criador de formulários**.

Para adicionar um objeto personalizado a partir da Barra de ferramentas controles

1 Na **Barra de ferramentas controles de formulários**, selecione o botão **Exibir Classes**.

2 Na lista de bibliotecas de classes registradas, escolha a biblioteca que contém o controle que você deseja adicionar ao formulário.

A barra de ferramentas receberá os controles da biblioteca selecionada.

Biblioteca de classes definida pelo usuário adicionada ao submenu Exibir classes

3 Clique sobre o controle desejado e arraste para dimensioná-lo no formulário.

Observação É possível remover uma biblioteca de classes visuais do menu da barra de ferramentas Exibir classes, selecionando a biblioteca na lista Selecionados, da guia **Controles** da caixa de diálogo **Opções** e ,em seguida, clicando sobre **Remover**.

Quando são adicionados objetos a um formulário com base em qualquer outra coisa que não sejam as classes base do Visual FoxPro, um caminho relativo para a biblioteca de classes (arquivo .VCX) é armazenado no arquivo .SCX do formulário. Se você mover o formulário ou a biblioteca de classes para uma localização diferente, o Visual FoxPro exibirá uma caixa de diálogo durante a tentativa de execução do formulário para que seja possível localizar a biblioteca de classes manualmente.

Determinando os controles de um formulário

Para determinar a quantidade de controles do formulário, você pode utilizar a propriedade ControlCount. A propriedade Controls[n] do formulário permite que você faça referência a cada um dos controles do formulário. O programa a seguir imprime a propriedade Name de todos os controles no formulário ativo.

```
ACTIVATE SCREEN  && para imprimir na janela principal do Visual FoxPro
FOR nCnt = 1 TO Application.ActiveForm.ControlCount
   ? Application.ActiveForm.Controls[nCnt].Name
ENDFOR
```

Adicionando propriedades e métodos a um formulário

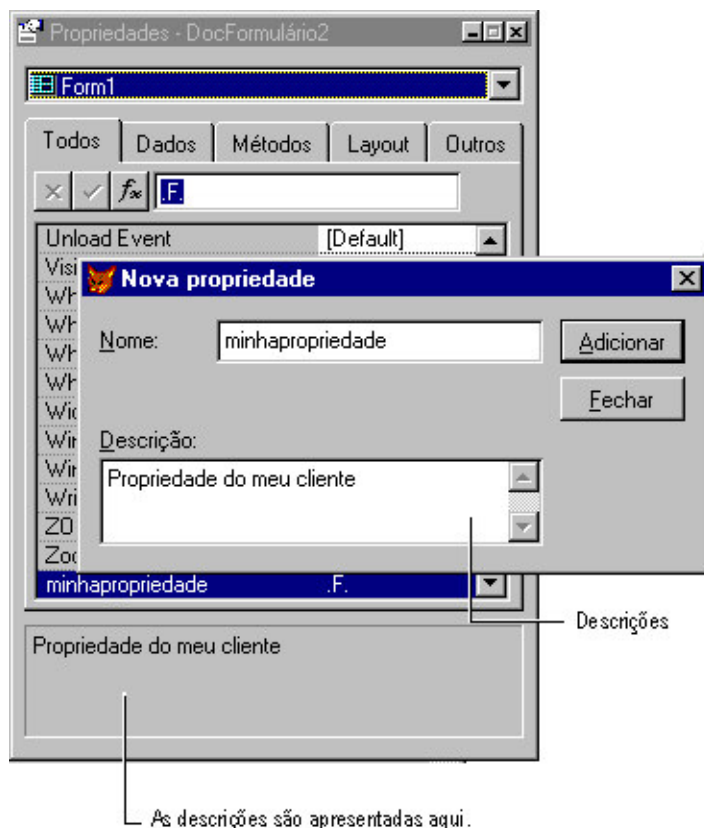
Pode-se adicionar o número de propriedades e métodos novos que desejar a um conjunto de formulários ou a um formulário que não faça parte de um conjunto. As propriedades armazenam valores; os métodos armazenam códigos de procedimento, que serão executados ao ser chamado o método. As novas propriedades e métodos são dimensionados para o formulário, e você pode fazer referência a eles da mesma forma que faz referência a outras propriedades e métodos do formulário.

Criando novas propriedades

Se você tiver um conjunto de formulários, as propriedades e os métodos adicionados ao **Criador de formulários** serão dimensionados para esse conjunto. Se não houver um conjunto de formulários, as propriedades e os métodos estarão dimensionados no formulário.

Para adicionar uma nova propriedade ao formulário

1. No menu **Formulário**, selecione a opção **Nova propriedade**.
2. Na caixa de diálogo **Nova propriedade**, digite o nome da propriedade. Você pode também incluir uma descrição da propriedade que será exibida na parte inferior da janela **Propriedades**.

Adicionando uma propriedade a um formulário

Criando uma propriedade Array

As propriedades Array estão dimensionadas no formulário como qualquer outra propriedade, mas podem ser manipuladas através dos comandos e das funções de matrizes do Visual FoxPro.

Para criar a propriedade Array

1. Adicione uma nova propriedade ao formulário.
2. Na caixa **Nome** da caixa de diálogo **Nova propriedade**, digite o nome da propriedade da matriz e inclua o seu tamanho e dimensões.

Por exemplo, você poderia digitar **arrayprop[10,2]** na caixa **Nome** da caixa de diálogo **Nova propriedade** para criar uma matriz bidimensional com 10 linhas.

As propriedades de matriz são somente para leitura no modo de criação, mas você pode controlar, redimensionar e atribuir valores aos elementos de propriedade da matriz em tempo de execução. Para obter um exemplo de utilização de propriedades de matrizes, consulte Gerenciando múltiplas instâncias de um formulário ainda neste capítulo.

Criando novos métodos

É possível adicionar ao formulário métodos que podem ser chamados da mesma maneira que os métodos de classes de formulários.

Para criar novos métodos para formulários

1. No menu **Formulário**, selecione a opção **Novo método**.
2. Na caixa de diálogo **Novo método**, digite o nome do método. Opcionalmente, você pode incluir uma descrição do método.

Os métodos definidos pelo usuário são chamados da mesma maneira que os métodos de classes principais, utilizando a sintaxe a seguir:

NomeObjeto.NomeMétodo
Seu método pode também aceitar parâmetros e retornar valores. Neste caso, o método deve ser chamado em uma instrução de atribuição:

cVariável = *NomeObjeto.NomeMétodo* (*cParâmetro*, *nParâmetro*)

Incluindo constantes predefinidas

Para utilizar constantes predefinidas nos seus métodos, você pode incluir um arquivo de cabeçalho no formulário ou conjunto de formulários utilizando #INCLUDE. Os arquivos de cabeçalho normalmente contêm constantes de tempo de compilação definidas com a diretiva de pré-processador #DEFINE.

Para incluir um arquivo em um formulário

1. No menu **Formulário**, selecione a opção **Incluir arquivo**.
2. Na caixa de diálogo **Incluir arquivo**, especifique o arquivo na caixa de texto **Incluir arquivo**.

   – Ou –

   Escolha o botão de reticências para abrir a caixa de diálogo **Incluir** e escolha o arquivo.
3. Escolha **OK**.

Manipulando objetos

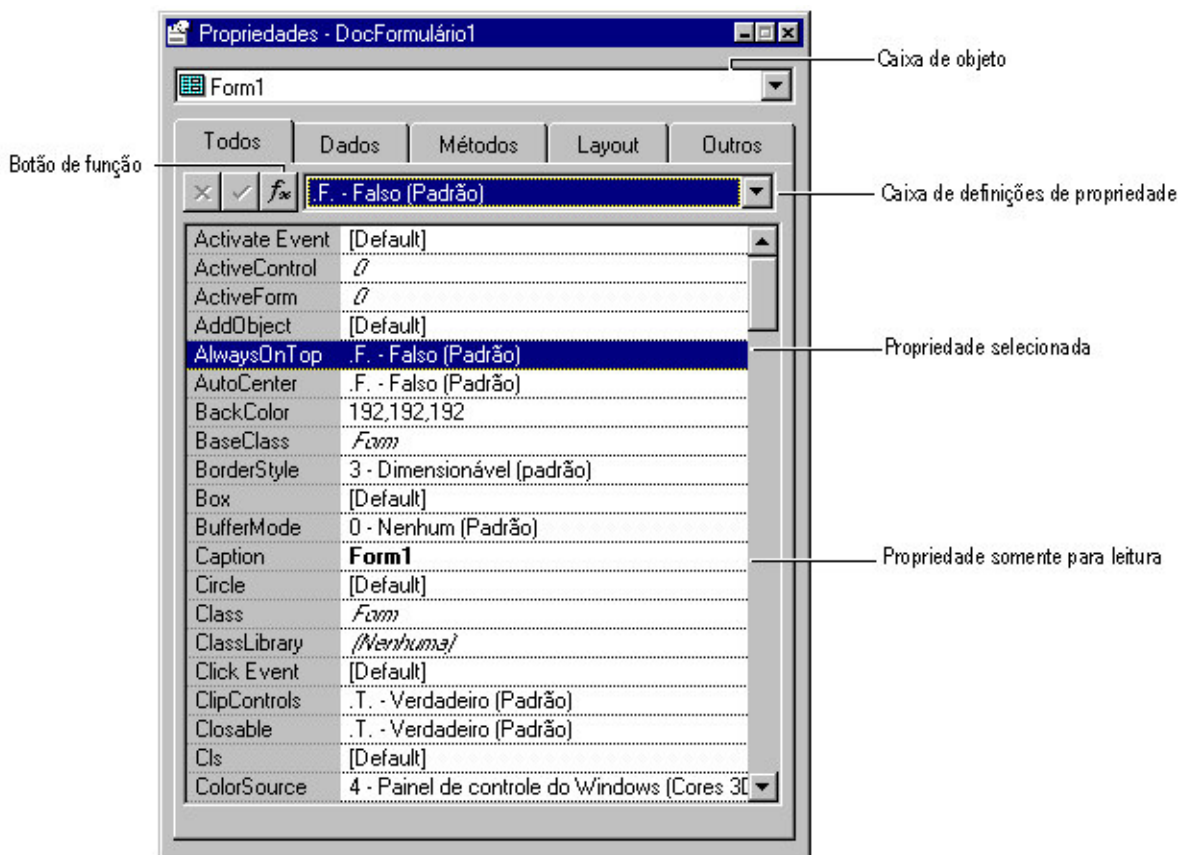
Há várias maneiras de manipular objetos durante a criação:

- Definir o tamanho e a posição dos objetos, arrastando-os na janela do **Criador de formulários**.
- Alinhar os controles através dos recursos de alinhamento da **Barra de ferramentas layout** ou das opções do menu **Formatar**.

- Definir as cores, escolhendo as cores de primeiro e de segundo planos na **Barra de ferramentas paleta de cores**.
- Definir as propriedades na janela **Propriedades**. O principal ponto de controle de todos os objetos do seu formulário é a janela **Propriedades**.

Definindo propriedades durante a criação

A janela **Propriedades** é aberta exibindo as propriedades ou os eventos do objeto selecionado. Se mais de um objeto estiver selecionado, as propriedades que os objetos têm em comum serão exibidas na janela **Propriedades**. Para editar as propriedades ou os eventos de um outro objeto, selecione o objeto desejado na caixa **Objeto** ou selecione um controle diferente no formulário.

Janela Propriedades

Para definir propriedades

1 Na Janela **Propriedades**, selecione uma propriedade na lista **Propriedades e eventos**.

2 Na caixa **Definições de propriedades**, digite ou escolha a definição desejada para a propriedade selecionada.

Observação As propriedades que forem somente para leitura durante a criação, como a propriedade Class de um objeto, serão exibidas na lista **Propriedades e eventos**, na janela **Propriedades**, em itálico.

Se a propriedade exigir um valor de caractere, não será necessário delimitá-lo entre aspas. Se você desejar que a legenda do formulário seja CLIENTE, digite **CLIENTE** na caixa **Definições de propriedades**. Se você desejar que a legenda do formulário seja "CLIENTE", com as aspas aparecendo no título da janela, digite **“CLIENTE”** na caixa **Definições de propriedades**.

Definindo propriedades com expressões

Você também pode definir propriedades para os resultados de expressões ou funções através da janela **Propriedades**.

Para definir propriedades com expressões

- Na janela **Propriedades**, escolha o botão **Função** para abrir o **Construtor de expressões.**

  – Ou –

- Na caixa **Definições de propriedades**, digite **=** seguido pela expressão.

Por exemplo, você pode definir a propriedade Caption do formulário de modo a apresentar a tabela ativa no momento em que o formulário for executado, digitando **=ALIAS( )** na caixa **Definições de propriedades**.

A expressão de propriedade é avaliada quando esta é definida na janela **Propriedades** e quando o objeto é inicializado em tempo de execução ou durante a criação. Uma vez criado o objeto, a definição de propriedade não se altera até que você ou o um usuário faça isso explicitamente.

Resolvendo problemas Se você definir uma propriedade como o resultado de uma função definida pelo usuário, essa função será avaliada quando você definir a propriedade ou modificar ou executar o formulário. Se houver algum erro na função definida pelo usuário, você poderá não conseguir abrir seu formulário.

Você pode também definir a propriedade para a função definida pelo usuário no evento Init do objeto, como no exemplo a seguir:

```
THIS.Caption = myfunction( )
```

Se houver um erro na função definida pelo usuário, ainda assim você não poderá executar o formulário dessa maneira, mas poderá modificá-lo.

Definindo o comportamento do formulário

Quando você estiver criando um formulário no **Criador de formulários**, esse formulário estará vivo: as alterações visuais e de comportamento que você fizer serão refletidas imediatamente sobre o mesmo, exceto a definição da propriedade Visible como falsa (.F.). Se você definir a propriedade WindowState com o valor 1 – Minimizado ou 2 – Maximizado o formulário presente no **Criador de formulários** refletirá imediatamente essa definição. Se você definir a propriedade Movable como falsa (.F.), o usuário não conseguirá movimentar o formulário em tempo de execução e você também não conseguirá movê-lo durante a criação,. Talvez você precise definir o funcionamento do seu formulário e adicionar todos os controles necessários antes de definir algumas das propriedades que determinam o comportamento do formulário.

As propriedades de formulário a seguir são normalmente estabelecidas durante a criação, definindo a aparência e o comportamento do formulário.

Propriedade	Descrição	Padrão
AlwaysOnTop	Define se o formulário ficará sempre por cima das outras janelas que estiverem abertas.	Falso (.F.)
AutoCenter	Determina se o formulário será centralizado automaticamente na janela principal do Visual FoxPro ou na área de trabalho quando for inicializado.	Falso (.F.)
BackColor	Determina a cor da janela do formulário.	255,255,255
BorderStyle	Define se o formulário não terá borda, se terá uma borda com linha única, com linha dupla ou se terá uma borda de sistema. Se BorderStyle for igual a 3–Sistema, o usuário poderá redimensionar o formulário.	3

Caption	Define o texto exibido na barra de título do formulário.	Form1
Closable	Define se o usuário poderá fechar o formulário com um clique duplo na caixa **Fechar**.	Verdadeiro (.T.)
DataSession	Define se as tabelas do formulário serão abertas nas áreas de trabalho que são acessíveis globalmente ou em áreas restritas ao formulário.	1
MaxButton	Define se o formulário terá ou não um botão de maximizar.	Verdadeiro (.T.)
MinButton	Define se o formulário terá ou não um botão de minimizar.	Verdadeiro (.T.)
Movable	Define se o formulário poderá ou não ser movido para outra localização na tela.	Verdadeiro (.T.)
ScaleMode	Define se a unidade de medida das propriedades de tamanho e posição do objeto será definida em foxels ou pixels.	Determinado pelas definições na caixa de diálogo **Opções**.
ShowWindow	Define se a janela é do tipo filho (na tela), flutuante ou de nível máximo.	0 - Na Tela
WindowState	Define se o formulário será minimizado (somente no Windows), maximizado ou normal.	0 - Normal
WindowType	Define se o formulário será não modal (o padrão) ou modal. Se o formulário for modal, o usuário deverá fechá-lo para poder acessar qualquer outro elemento da interface com o usuário do seu aplicativo.	0–Não Modal

Você pode utilizar a propriedade LockScreen para fazer com que o ajuste em tempo de execução das propriedades de layout de controle fique mais preciso.

Atribuindo ícones a formulários

No Visual FoxPro para Windows, você pode atribuir um ícone ao formulário; o ícone será exibido quando a janela for minimizadas no Windows NT® e na barra de título do Windows 95. Para atribuir um ícone a um formulário, defina a propriedade Icon do formulário como nome de um arquivo .ICO.

Para atribuir um ícone a um formulário

1 Abra o formulário.
2 Abra a janela **Propriedade**.
3 Defina a propriedade **Icon** para o arquivo .ICO que você deseja exibir.

Editando códigos de eventos e de métodos

Eventos são ações do usuário, como cliques ou movimentos do mouse, ou ações do sistema, como a seqüência do relógio do sistema. Métodos são procedimentos associados ao objeto e que são chamados especificamente, de forma programática. Para uma discussão sobre eventos e métodos, consulte o capítulo 3, Programação orientada a objetos. Você pode especificar o código a ser processado quando um evento for disparado ou quando um método for chamado.

Para editar códigos de eventos ou de métodos

1 No menu **Exibir**, selecione a opção **Código**.
2 Selecione o evento ou o método na caixa **Procedimento**.
3 Na janela **Editar**, escreva o código que deseja que seja processado quando ocorrer um evento ou quando algum método for chamado.

Por exemplo, você poderia ter um botão de comando em um formulário, com a legenda “Sair”. No evento Click correspondente a esse botão inclua a linha:

```
THISFORM.Release
```

Dica Para mover-se entre os procedimentos na janela **Edição de código**, pressione PAGE DOWN ou PAGE UP.

Quando o usuário clicar sobre o botão de comando, o formulário será removido da tela e da memória. Se você não desejasse retirar o formulário da memória, poderia incluir a linha a seguir no evento de clique:

```
THISFORM.Hide
```

Observação Se o código associado ao evento Init de um conjunto de formulários, formulário ou de qualquer objeto de qualquer formulário do conjunto de formulários retornar falso (.F.), o formulário não será criado.

Salvando formulários

Você precisa salvar seu formulário antes de poder executá-lo. Se você tentar fechar o **Criador de formulários** sem que o formulário tenha sido salvo, o Visual FoxPro emitirá um aviso para que você salve ou descarte as alterações que tiver feito.

Para salvar o formulário

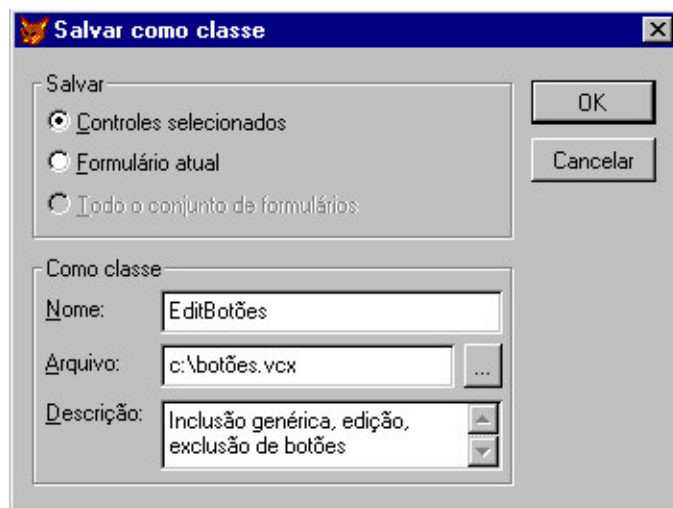
- No **Criador de formulários**, escolha a opção **Salvar** no menu **Arquivo**.

Salvando formulários e controles como classes

É possível salvar um formulário, ou um subconjunto dos controles do formulário, como uma definição de classe. Se você pretende criar subclasses baseadas no formulário ou reutilizar os controles em outros formulários, salve o formulário como uma definição de classe.

Para salvar o formulário ou os controles selecionados como uma definição de classe

1 No menu **Arquivo**, selecione a opção **Salvar como classe**.
2 Na caixa de diálogo **Salvar como classe**, selecione **Formulário atual** ou **Controles selecionados**.

Caixa de diálogo Salvar como classe

3 Na caixa **Nome**, digite um nome para a classe.

4 Na caixa **Arquivo**, digite um nome de arquivo onde a classe deverá ser armazenada.

5 Escolha **OK**.

Caso você não especifique uma extensão para o nome do arquivo, será atribuída a extensão padrão .VCX ao ser gravado o arquivo. Uma vez que o formulário tenha sido salvo como uma definição de classe, você poderá modificá-lo com o comando MODIFYCLASS. Para obter maiores informações sobre a criação de classes, consulte o capítulo 3, Programação orientada a objetos.

Executando um formulário

Pode-se executar um formulário diretamente da interface ou a partir de um código de programa.

Executando um formulário em modo interativo

Há várias maneiras de executar o formulário que você criou.

Se você estiver trabalhando no **Criador de formulários**, poderá testar o formulário, clicando sobre o botão Executar da **Barra de ferramentas criador de formulários**. Para abrir novamente o formulário no **Criador de formulários**, feche o formulário ou escolha o botão **Modificar formulário** na barra de ferramentas.

Pode-se também executar o formulário a partir de um projeto ou utilizando a linguagem de programação.

Para executar o formulário interativamente

- No **Gerenciador de projetos**, selecione o formulário e escolha **Executar**.

  – Ou –

- Digite **DO FORM** na janela **Comando**.

Pode-se ainda executar o formulário através da opção **Executar** do menu **Programa**. Basta selecionar **Formulário** na caixa **Tipo de arquivo**, selecionar o formulário e escolher a opção **Executar**.

Executando um formulário a partir de um programa

Para executar o formulário através da programação, inclua o comando **DO FORM** no código associado a um evento, no código de método, em um programa ou em um procedimento.

Nomeando o objeto formulário

Como padrão, quando for utilizado o comando **DO FORM**, o nome do objeto formulário será o mesmo do arquivo .SCX. Por exemplo, a linha de código a seguir executa CUSTOMER.SCX. O Visual FoxPro cria automaticamente uma variável de objeto para o formulário chamado `customer`:

```
DO FORM Customer
```

Para nomear o objeto formulário

- Utilize a cláusula NAME do comando DO FORM.

Por exemplo, os comandos a seguir executam um formulário, criando dois nomes de variáveis para o objeto formulário:

```
DO FORM Customer NAME frmCust1
DO FORM Customer NAME frmCust2
```

Manipulando o objeto formulário

Se você emitir o comando DO FORM na janela **Comando**, o objeto formulário será associado a uma

variável pública. Você poderá acessar o objeto formulário através do nome da variável. Por exemplo, os comandos a seguir, emitidos na janela **Comando**, abrem um formulário chamado `Customer` e mudam sua legenda.

```
DO FORM Customer
Customer.Caption = "Oi"
```

Se depois você emitir o comando abaixo na janela **Comando**, será exibido `O` na janela de saída ativa, indicando que `Customer` é um objeto:

```
? TYPE("Customer")
```

Se você emitir o comando DO FORM em um programa, o objeto formulário estará no escopo desse programa. Se o programa ou procedimento completar sua execução, o objeto terminará, mas o formulário permanecerá visível. Por exemplo, você poderia executar o programa a seguir:

```
*formtest.prg
DO FORM Customer
```

A pós executar o programa, o formulário permaneceria visível e todos os controles do formulário ficariam ativos, mas `TYPE("Customer")` retornaria `U`, indicando que `Customer` seria uma variável indefinida. O comando a seguir, emitido na janela **Comando**, geraria um erro:

```
Customer.Caption = "Oi"
```

Contudo, você pode acessar o formulário através das propriedades ActiveForm, Forms e FormCount do objeto do aplicativo.

Definindo o escopo do formulário como a variável do objeto formulário

A palavra-chave LINKED do comando DO FORM permite que você vincule o formulário ao respectivo objeto formulário. Se você incluir a palavra-chave LINKED, quando a variável associada ao objeto formulário sair do escopo, o formulário será liberado.

Por exemplo, o comando a seguir cria um formulário vinculado à variável de objeto `frmCust2`:

```
DO FORM Customer NAME frmCust2 LINKED
```

Quando `frmCust2` for liberada, o formulário será fechado.

Fechando um formulário ativo

Para permitir que o usuário feche o formulário ativo clicando duas vezes sobre a caixa de controle ou selecionando a opção **Fechar** no menu **Controle do formulário**, defina a propriedade Closable do formulário.

Para permitir que um usuário feche o formulário ativo

- Na janela **Propriedades**, defina a propriedade Closable como verdadeiro (.T.).

  – Ou –

- Utilize o comando RELEASE.

Por exemplo, você pode fechar e liberar o formulário `frmCustomer` emitindo o seguinte comando em um programa ou na janela **Comando**:

```
RELEASE frmCustomer
```

Você pode ainda permitir que o usuário feche e libere o formulário incluindo o seguinte comando no código de evento Click de um controle, como um botão de comando com a legenda “Sair”.

```
THISFORM.Release
```

Você pode também utilizar o comando RELEASE no código associado a um objeto no formulário, mas qualquer código incluído no método Release não será executado.

Resolvendo problemas Ao liberar um formulário, você libera também da memória a variável de objeto criada para ele. Só há uma variável para um conjunto de formulários, de modo que você não pode liberar os formulários de um conjunto sem que este seja liberado. Se desejar liberar o conjunto de formulários, você poderá utilizar `RELEASE THISFORMSET`. Se desejar remover um formulário da

tela para que o usuário não possa mais vê-lo nem interagir com o mesmo, você poderá utilizar `THISFORM.Hide`.

Definindo propriedades em tempo de execução

O modelo de objeto do Visual FoxPro lhe oferece um grande grau de controle sobre as propriedades em tempo de execução.

Fazendo referência a objetos na hierarquia de objetos

Para manipular objetos, você precisa identificá-los em relação à hierarquia de recipientes. No nível mais alto da hierarquia de recipientes (o conjunto de formulários ou formulário), você precisa fazer referência à variável de objeto. A não ser que utilize a cláusula NAME do comando DO FORM, a variável de objeto possuirá o mesmo nome que o arquivo .SCX.

As propriedades são manipuladas fazendo-se referência à variável de objeto, o controle e a propriedade, separados por um ponto (.):

variávelobjeto.[*formulário*.]*controle*.*propriedade* = *Definição*
A tabela a seguir lista as propriedades ou palavras-chave que facilitam a referência a objetos na hierarquia de objetos:

Propriedade ou palavra-chave	Referência
ActiveControl	Controle do formulário ativo que está sendo destacado
ActiveForm	Formulário ativo no momento
ActivePage	Página ativa do formulário ativo no momento
Parent	Recipiente imediato do objeto
THIS	O objeto ou um procedimento ou evento do objeto
THISFORM	Formulário que contém o objeto
THISFORMSET	Conjunto de formulários que contém o objeto

Por exemplo, para alterar a legenda de um botão de comando no formulário `frmCust` em um conjunto de formulários armazenado em CUSTVIEW.SCX, utilize o comando a seguir em um programa ou na janela **Comando**:

```
CustView.frmCust.cmdButton1.Caption = "Editar"
```

Utilize as palavras-chave THIS, THISFORM e THISFORMSET para fazer referência a objetos de dentro de um formulário. Por exemplo, para alterar a legenda de um botão de comando quando o mesmo for clicado, inclua o comando a seguir no código de evento Click do botão de comando:

```
THIS.Caption = "Editar"
```

A tabela a seguir apresenta alguns exemplos da utilização de THISFORMSET, THISFORM, THIS e Parent para definir propriedades de objetos:

Comando	Onde incluir o comando
`THISFORMSET.frm1.cmd1.Caption = 'OK'`	No código de evento ou de método de qualquer controle de qualquer formulário do conjunto de formulários, com exceção de `frm1`.
`THISFORM.cmd1.Caption = 'OK'`	No código de evento ou de método de qualquer controle, com exceção de `cmd1` no mesmo formulário em que estiver `cmd1`.
`THIS.Caption = 'OK'`	No código de evento ou de método do controle cuja legenda deverá ser

	alterada.
`THIS.Parent.BackColor = RGB(192,0,0)`	No código de evento ou de método de um controle de um formulário. Este comando altera a cor do segundo plano do formulário para vermelho-escuro.

Definindo propriedades em tempo de execução com expressões

Também é possível definir propriedades em tempo de execução utilizando-se expressões ou funções.

Para definir propriedades com expressões em tempo de execução

- Atribua uma expressão à propriedade.

  – Ou –

- Atribua à propriedade o resultado de uma função definida pelo usuário.

Por exemplo, você poderia definir a legenda de um botão como Editar ou Salvar, dependendo do valor de uma variável. Declare a variável no programa de chamada do seu formulário:

```
PUBLIC glEditing
glEditing = .F.
```

Em seguida, utilize uma expressão IIF na definição de Caption:

```
frsSet1.frmForm1.cmdButton1.Caption = ;
   IIF(glEditing = .F., "Editar", "Salvar")
```

Você poderia determinar o tamanho de um botão e definir a legenda utilizando expressões com campos de uma tabela:

```
* define a largura do botão como o comprimento de 'Call ' + nome e sobrenome
frmForm1.cmdButton1.Width = 5 + ;
   LEN(ALLTRIM(employee.first_name      + " " + employee.last_name))
* define a legenda do botão como 'Call ' + nome e sobrenome
frmForm1.cmdButton1.Caption = "Call " + ;
   ALLTRIM(employee.first_name + " " + employee.last_name)
```

Você poderia ainda definir a legenda com uma função definida pelo usuário:

```
frsSet1.frmForm1.cmdButton1.Caption = setcaption()
```

Definindo várias propriedades

É possível definir várias propriedades de uma só vez.

Para definir várias propriedades

- Utilize a estrutura WITH ... ENDWITH.

Por exemplo, para definir várias propriedades para uma coluna em uma grade de um formulário, você poderia incluir a instrução a seguir em qualquer código de evento ou método do formulário:

```
WITH THISFORM.grdGrid1.grcColumn1
   .Width = 5
   .Resizable = .F.
   .ForeColor = RGB(0,0,0)
   .BackColor = RGB(255,255,255)
   .SelectOnEntry = .T.
ENDWITH
```

Chamando métodos em tempo de execução

A sintaxe para chamar métodos de objetos é:

Pai.Objeto.Metodo

Uma vez que o objeto tenha sido criado, pode-se chamar os métodos desse objeto de qualquer

ponto do aplicativo. Os comandos a seguir fazem chamadas a métodos para exibir um formulário e passam o destaque para um botão de comando:

```
* conjunto de formulários salvo em MYF_SET.SCX
myf_set.frmForm1.Show
myf_set.frmForm1.cmdButton1.SetFocus
```

Para ocultar o formulário, emita este comando:

```
myf_set.frmForm1.Hide
```

Respondendo a eventos

O código que for incluído em um procedimento de evento será executado quando esse evento ocorrer. Por exemplo, o código que for incluído no procedimento do evento Click de um botão de comando será executado quando o usuário clicar sobre esse botão.

Chamar o código de procedimento associado a um evento não faz com que o evento ocorra. Por exemplo, a instrução a seguir faz com que o código do evento Activate de `frmPhoneLog` seja executado, mas não ativa o formulário:

```
frmPhoneLog.Activate
```

A chamada do método Show do formulário faz com que o mesmo seja exibido e ativado, e neste ponto o código do evento Activate é executado:

```
frmPhoneLog.Show
```

Exemplo de manipulação de objetos

O exemplo a seguir define propriedades e chama códigos de eventos de diversos objetos em um conjunto de formulários. O exemplo inclui dois formulários, frmLeft e frmRight, em um conjunto de formulários.

Exemplo de conjunto de formulários no Criador de formulários

As duas caixas de verificação e o botão de comando do formulário frmLeft possuem códigos de eventos associados. O nome da caixa de texto de frmLeft é `txtInput`.

Código de evento para os objetos de LeftForm

Objeto	Evento	Código
chkItalic	Click	`THISFORM.txtInput.FontItalic = ; THIS.Value`
chkBold	Click	`THIS.txtInput.FontBold = THIS.Value`
		`THISFORM.txtInput.Value = ""`

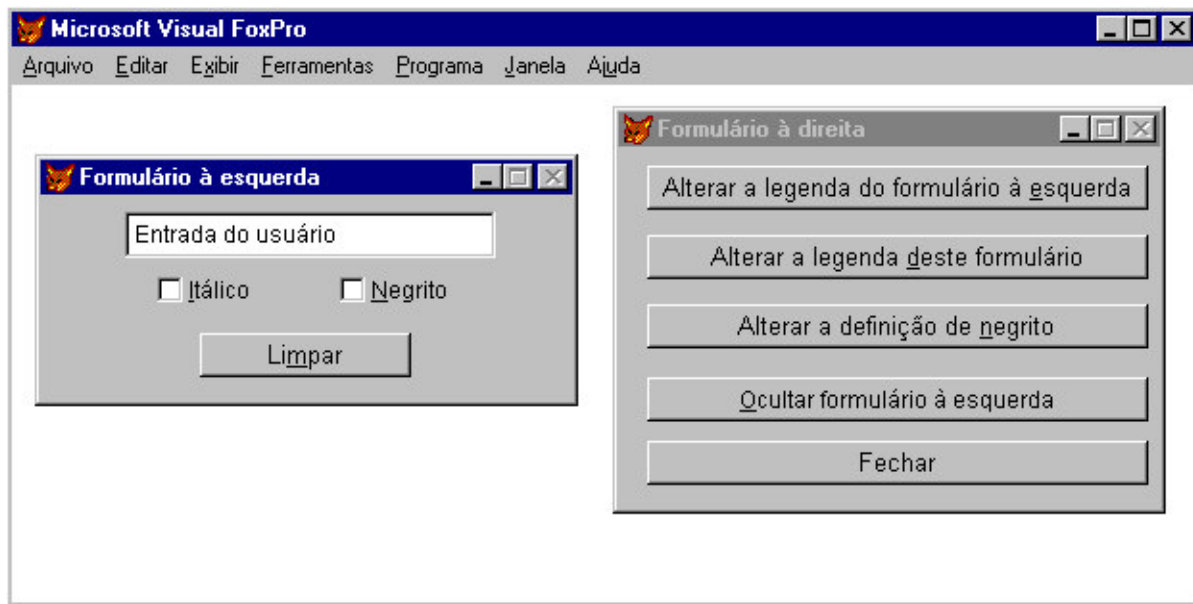
```
THISFORM.txtInput.FontBold = .F.
THISFORM.txtInput.FontItalic = .F.
THISFORM.chkItalic.Value = .F.
THISFORM.chkBold.Value = .F.
```

Definindo uma propriedade de outro controle no mesmo formulário

Você pode definir as propriedades de um controle de dentro do código de evento de outro, utilizando a palavra-chave THISFORM ou propriedade Parent. Os dois comandos a seguir serão executados quando o usuário clicar pela primeira vez sobre as caixas de verificação **Italic** e **Bold**, definindo as propriedades adequadas para a caixa de texto:

```
THISFORM.txtInput.FontItalic = .T.
THIS.Parent.txtInput.FontBold = .T.
```

Neste caso, THISFORM e THIS.Parent podem ser utilizados um no lugar do outro.

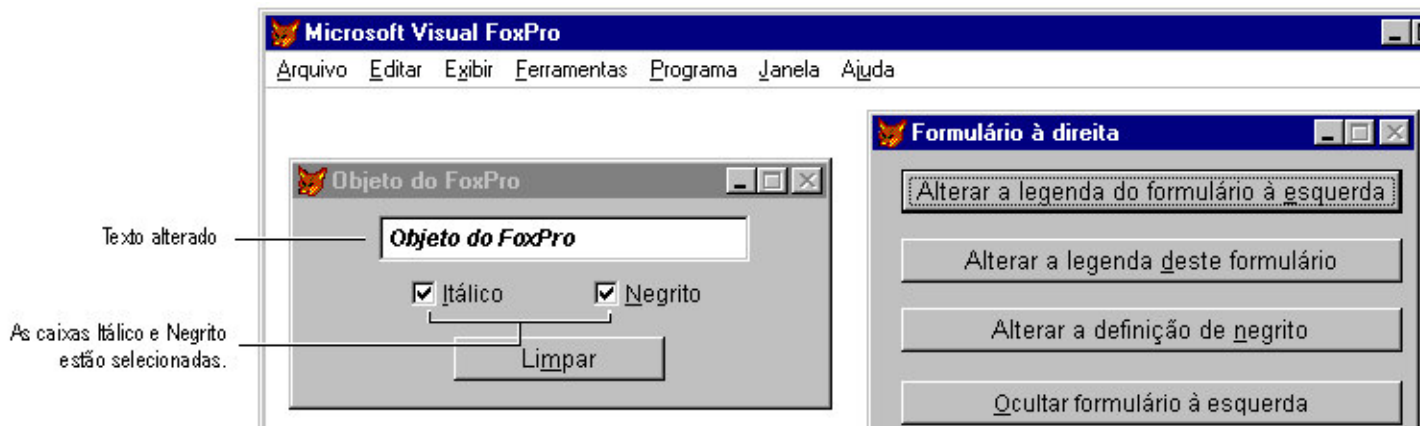
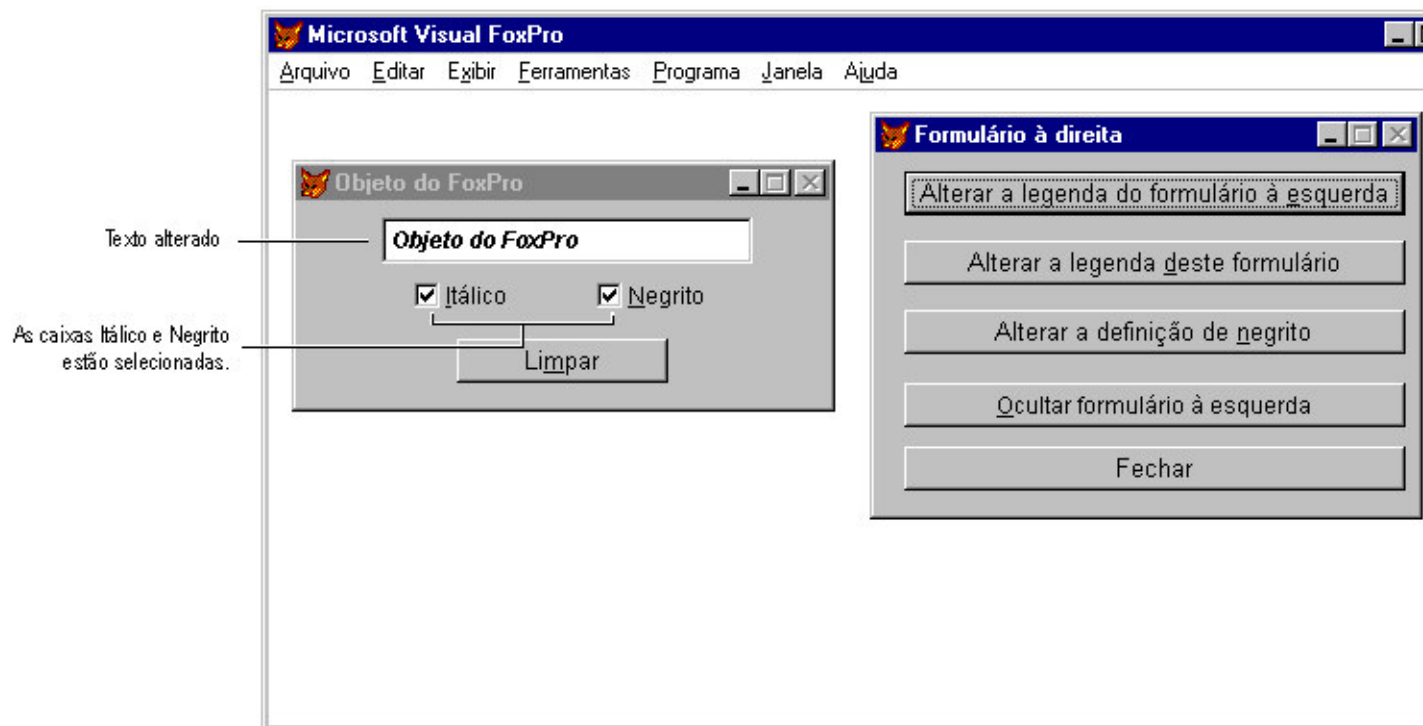
Exemplo de conjunto de formulários em tempo de execução

O código do evento Click de `cmdClear` utiliza THISFORM para redefinir os valores dos outros controles do formulário.

Definindo as propriedades de outro formulário

Pode-se também definir as propriedades de um formulário a partir de outro. Form2 contém cinco botões de comando. O primeiro botão do formulário possui o código a seguir no evento Click :

```
THISFORMSET.frmLeft.Caption = ;
   ALLTRIM(ThisFormSet.frmLeft.txtInput.Value)
```

Observe que deve-se fazer referência ao conjunto de formulários e ao formulário, ao serem definidas suas propriedades a partir de um outro formulário.

O usuário clica sobre o botão de comando Alterar a legenda do formulário à esquerda no formulário à direita

O código de evento Click do segundo botão de comando do formulário `frmRight` demonstra a definição de uma propriedade de um formulário a partir de um objeto no formulário:

```
THISFORM.Caption = ;
   ALLTRIM(ThisFormSet.frmLeft.txtInput.Value)
```

Se o usuário clicar sobre este botão, a legenda do formulário frmRight será alterada de acordo com o valor que estiver na caixa de texto do formulário frmLeft.

Acessando objetos em formulários diferentes

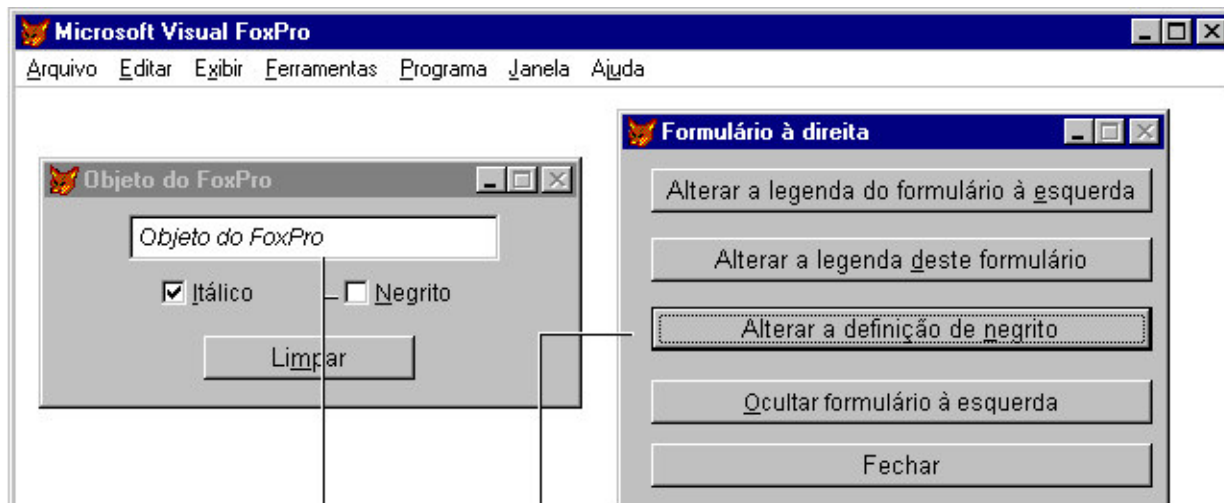
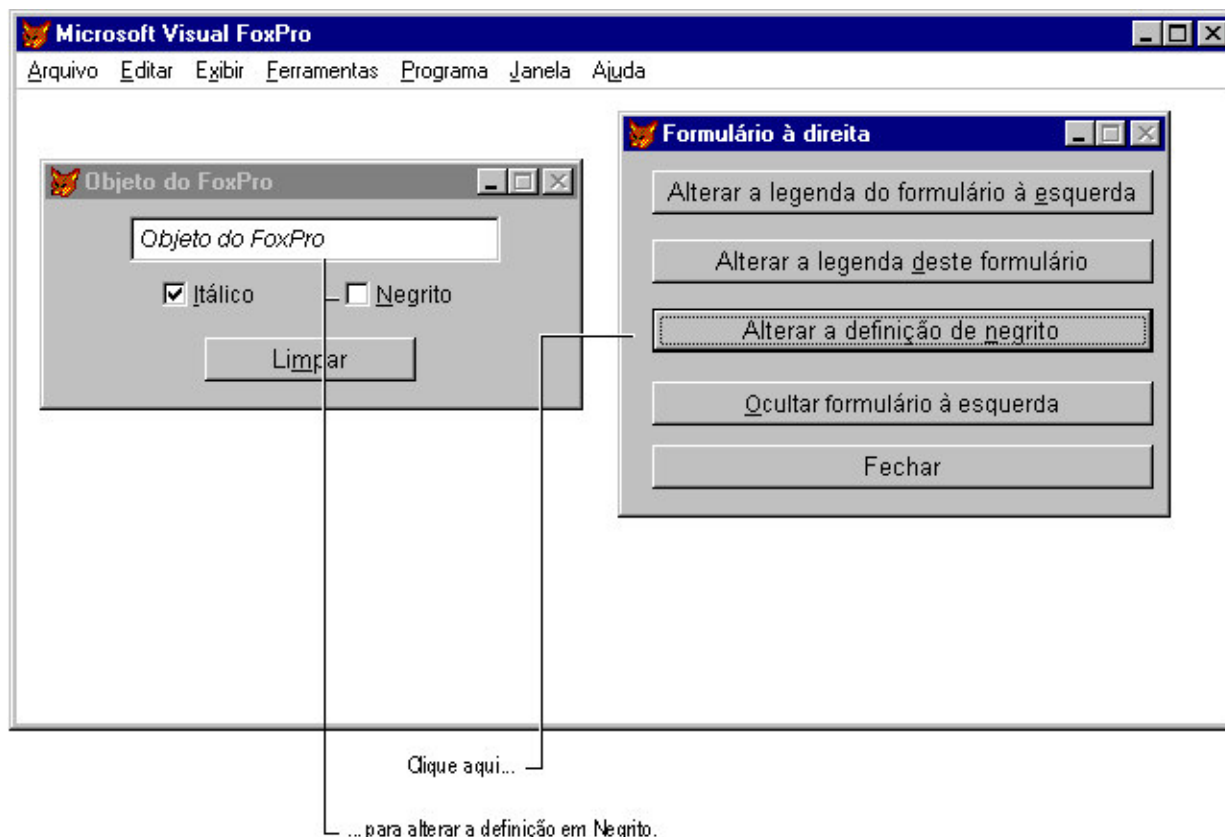
O código a seguir no evento Click do botão de comando Alterar a definição de negrito altera o valor da caixa de verificação **Negrito** no formulário frmLeft e chama o código de evento associado a esse controle.

```
THISFORMSET.frmLeft.chkBold.Value = ;
   NOT THISFORMSET.frmLeft.chkBold.Value
THISFORMSET.frmLeft.chkBold.InteractiveChange
```

A última linha do exemplo chama o evento InteractiveChange de `chkBold`. Você poderia também chamar esse procedimento com o comando a seguir:

```
THISFORMSET.frmForm1.chkBold.InteractiveChange( )
```

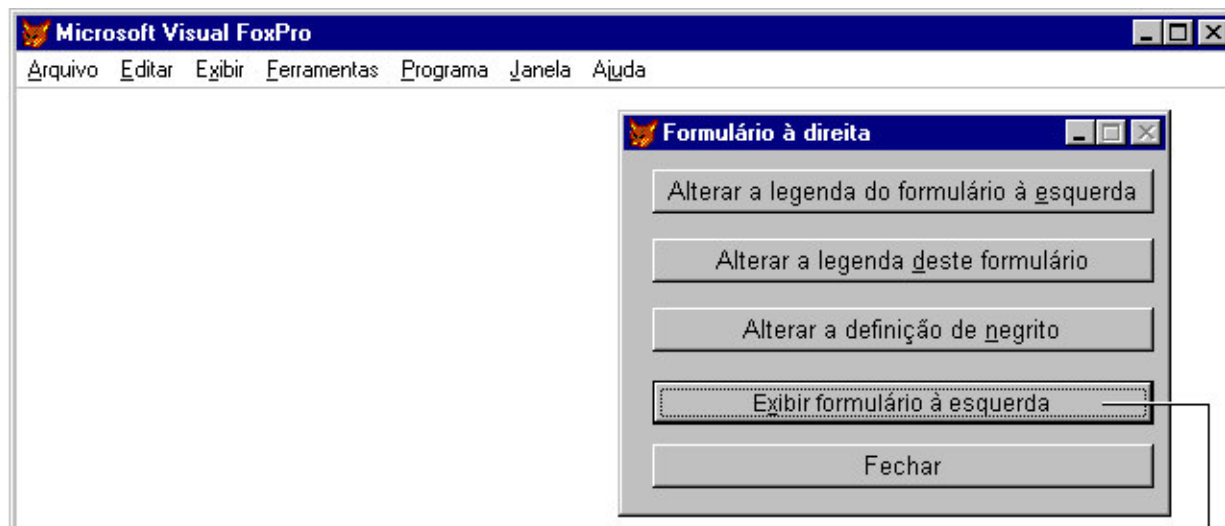
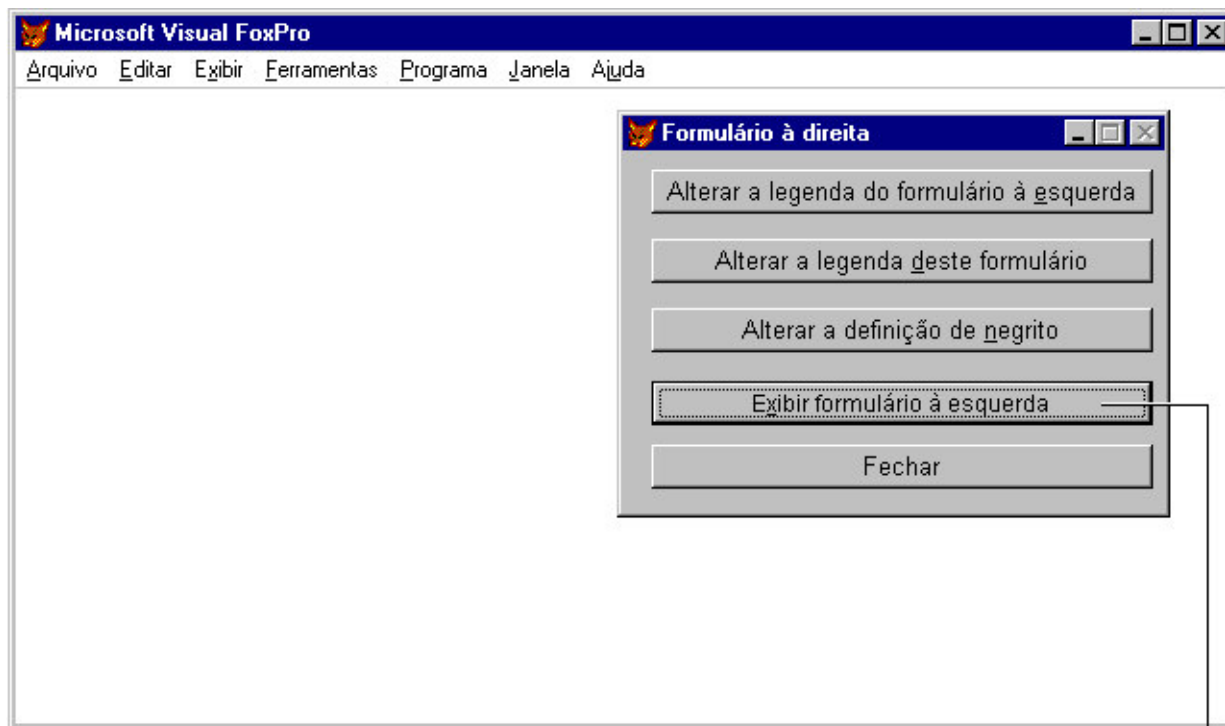
Se essa chamada de procedimento for omitida, o valor da caixa de verificação será alterado, mas a propriedade FontBold da caixa de texto nunca será alterada.

O usuário clica sobre o botão de comando Alterar a definição de negrito no formulário à direita

Verificando propriedades e chamando códigos de métodos de outro formulário

O código a seguir no evento Click do botão de comando **Ocultar formulário à esquerda** oculta ou exibe frmLeft, dependendo do valor da propriedade Visible e altera a legenda do botão como apropriado.

```
IF ThisFormSet.frmLeft.Visible
   ThisFormSet.frmLeft.Hide
   THIS.Caption = "Exibe formulário esquerdo "
ELSE
   ThisFormSet.frmLeft.Show
   THIS.Caption = "Ocultar formulário à esquerda"
ENDIF
```

Observe que a palavra-chave THIS foi utilizada dentro do código de evento de um controle para fazer referência às propriedades do controle.

O usuário clica sobre o botão de comando Ocultar formulário à esquerda no formulário à direita

O comando a seguir no evento Click do botão de comando **Sair** libera o conjunto de formulários, fazendo com que ambos os formulários sejam fechados:

```
RELEASE ThisFormSet
```

Gerenciando formulários

Os procedimentos a seguir descrevem tarefas comuns associadas ao gerenciamento dos formulários nos aplicativos.

Ocultando um formulário

Pode-se ocultar um formulário de modo que não fique visível para o usuário. Quando o formulário estiver oculto, o usuário não poderá interagir com o formulário, mas você continuará tendo total controle sobre eles através da programação.

Para ocultar um formulário

- Utilize o método Hide.

Por exemplo, no código associado ao evento Click de um botão de comando, você poderia incluir a linha de código a seguir:

```
THISFORM.Hide
```

Quando o usuário clicar sobre o botão de comando, o formulário permanecerá na memória, mas não ficará visível.

Liberando um formulário

Você pode permitir que o usuário libere um formulário ao terminar de interagir com ele. Ao liberar o formulário, você não poderá mais acessar suas propriedades e métodos.

Para liberar um formulário

- Utilize o método Release.

Por exemplo, no código associado ao evento Click de um botão de comando, você poderia incluir a linha de código a seguir:

```
THISFORM.Release
```

Quando o usuário clicar sobre o botão de comando, o formulário será fechado.

Passando parâmetros para um formulário

Algumas vezes é necessário passar parâmetros para formulários, ao executá-los para definir valores de propriedades ou especificar padrões operacionais.

Para passar um parâmetro para um formulário criado no Criador de formulários

1. Crie propriedades no formulário para armazenar os parâmetros, tais como ItemName e ItemQuantity.
2. No código do evento Init do formulário, inclua uma instrução PARAMETERS como:

   ```
   PARAMETERS cString, nNumber
   ```

3. No código de evento Init do formulário, atribua os parâmetros às propriedades, como neste exemplo:

   ```
   THIS.ItemName = cString
   THIS.ItemQuantity = nNumber
   ```

4. Ao executar o formulário, inclua uma cláusula WITH no comando DO FORM:

   ```
   DO FORM myform WITH "Bagel", 24
   ```

Retornando valores a partir de um formulário

Pode-se utilizar formulários no aplicativo para permitir que o usuário especifique valores.

Para retornar um valor de um formulário

1. Defina a propriedade WindowType do formulário como 1 para torná-lo modal.
2. No código associado ao evento UnLoad do formulário, inclua um comando RETURN com o valor de retorno.
3. No programa ou método que executar o formulário, inclua a palavra-chave TO no comando DO FORM.

   Por exemplo, se `FindCustID` for um formulário modal que retorna um valor de caractere, a linha de código a seguir armazenará o valor de retorno em uma variável chamada `cCustID`:

   ```
   DO FORM FindCustID TO cCustID
   ```

Para obter maiores informações, consulte RETURN e DO FORM na Ajuda.

Resolvendo problemas Se ocorrer algum erro, verifique se WindowType está definida como 1 (Modal).

Gerenciando múltiplas instâncias de um formulário

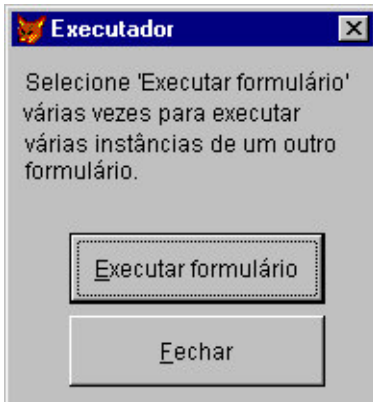
Pode haver várias instâncias da definição de classe ativa ao mesmo tempo. Por exemplo, você pode criar um formulário de pedido e ter diversos pedidos abertos no seu aplicativo, todos utilizando a mesma definição de formulário, mas exibidos e manipulados individualmente.

Quando houver várias instâncias do mesmo formulário, não se esqueça dos pontos básicos a seguir:

- Crie uma propriedade de matriz no formulário que está sendo executado, com a finalidade de armazenar as variáveis de objeto associadas a cada instância do formulário de múltiplas instâncias. A maneira mais fácil de manter o controle das variáveis de instância, quando você não souber com antecedência quantas delas irão existir, é utilizar uma matriz.
- Para o formulário que deverá ter várias instâncias, defina a propriedade DataSession como 2 – Sessão de Dados Privada. Uma sessão de dados privada fornece um conjunto separado de áreas de trabalho para cada instância do formulário, de modo que as tabelas selecionadas e as posições de ponteiro de registro sejam todas independentes.

O exemplo a seguir fornece um código que demonstra a criação de várias instâncias de um formulário. Para fins de rapidez, este código não é otimizado; sua intenção é apenas apresentar os conceitos.

O formulário a seguir executa várias instâncias:

Formulário acionador

Definição de propriedades para LAUNCH.SCX

Objeto	Propriedade	Definição
frmLaunch	aForms[1]	" "

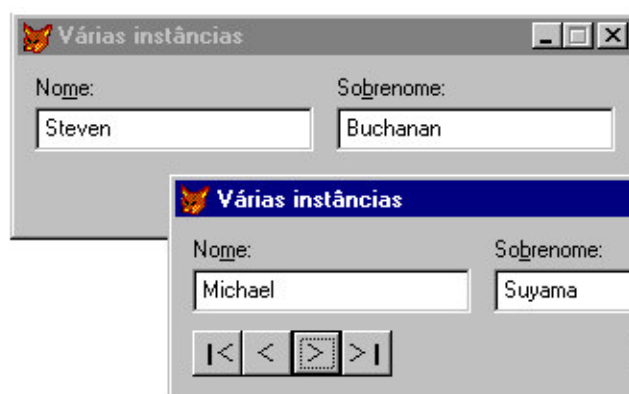
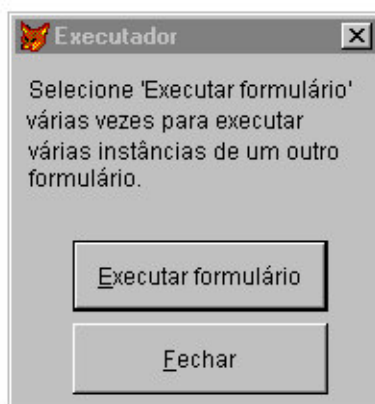
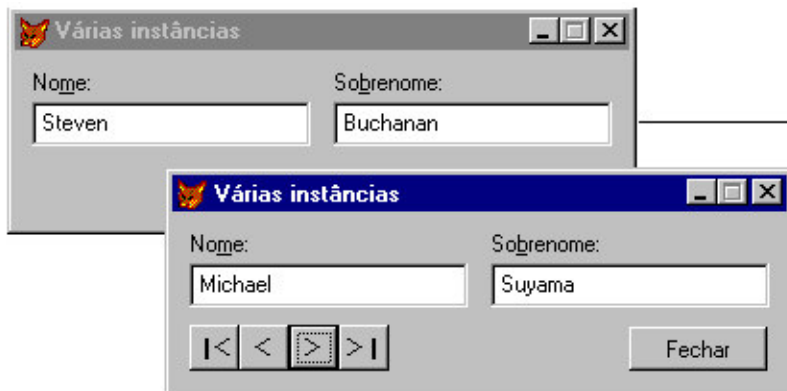
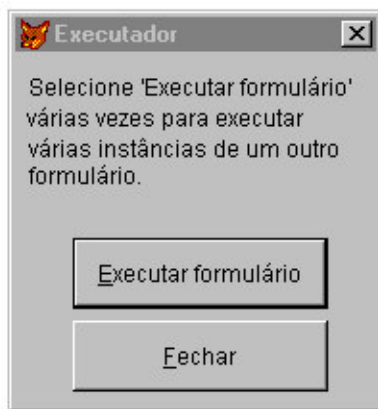
Código de evento para LAUNCH.SCX

Objeto	Evento	Código
cmdQuit	Click	`RELEASE THISFORM`

cmdLaunch	Click	`nInstance = ALEN(THISFORM.aForms)`
		`DO FORM Multi ;`
		`  NAME THISFORM.aForms[nInstance] ;`
		`  LINKED`
		`DIMENSION ;`
		`  THISFORM.aForms[nInstance + 1]`

Para refinar o código apresentado neste exemplo, você poderia gerenciar a matriz dos objetos formulário de forma que os elementos vazios da matriz reutilizados como formulários fossem fechados, e fossem abertos novos formulários, em vez de sempre redimensionar a matriz e aumentar o número de elementos em uma unidade.

O formulário que pode ter várias instâncias é MULTI.SCX. O ambiente de dados para este formulário contém a tabela Employee.

Várias instâncias de MULTI.SCX

Definição de propriedades para MULTI.SCX

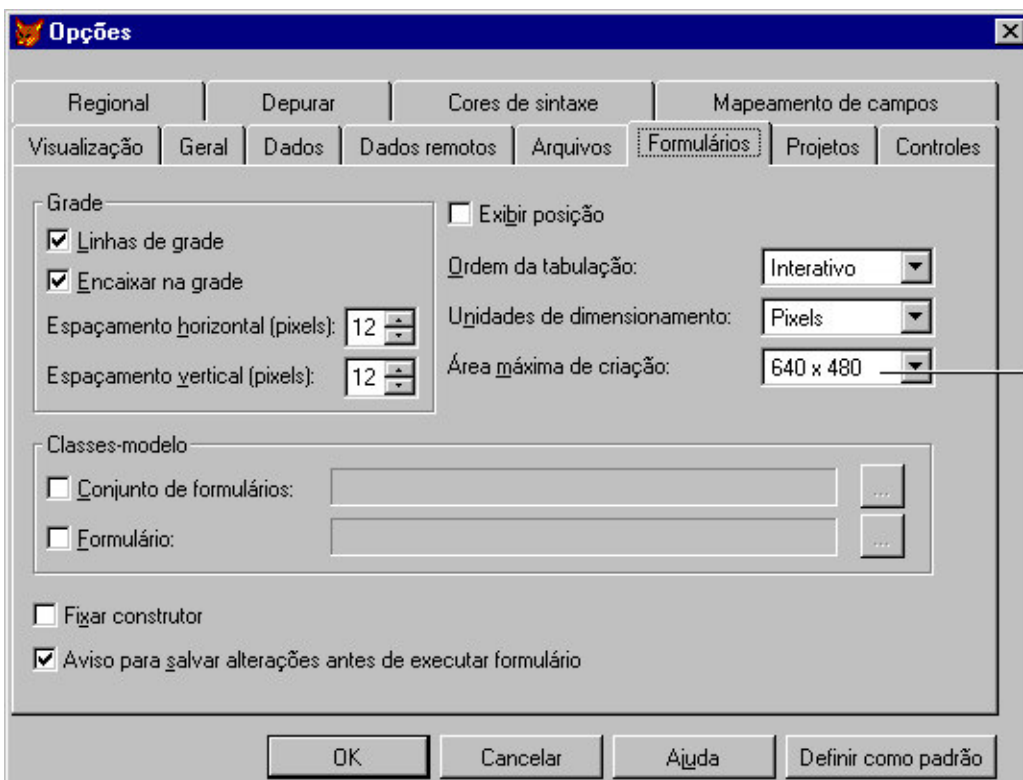
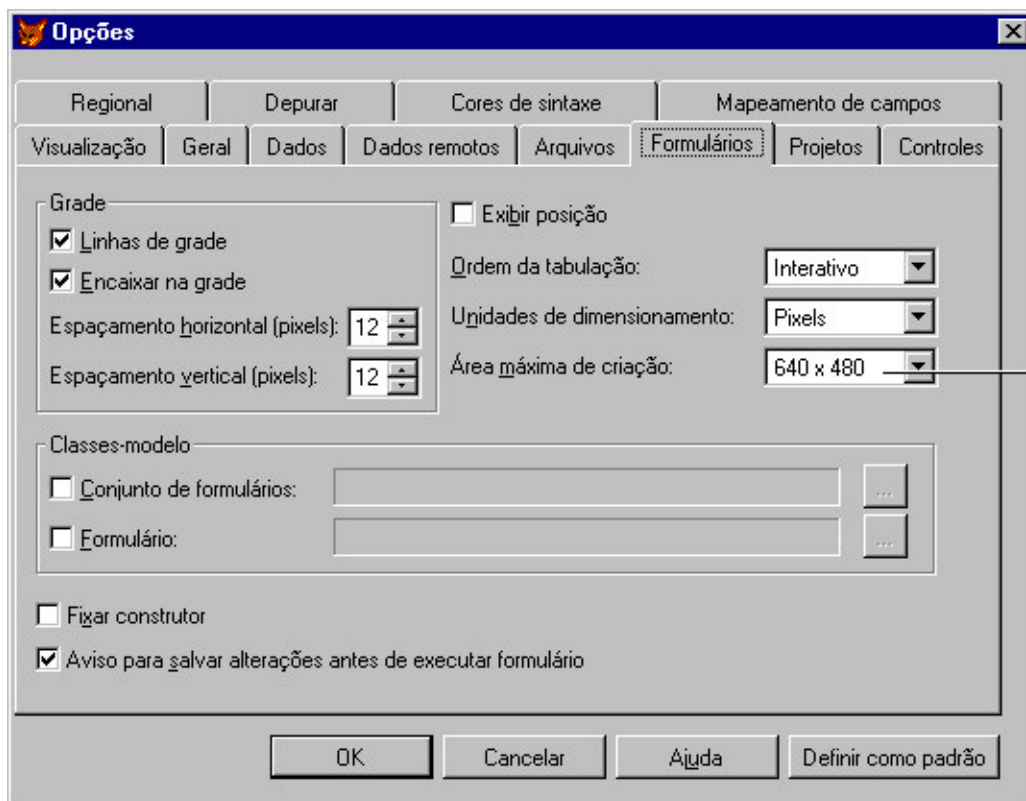
Objeto	Propriedade	Definição
txtFirstname	ControlSource	`Employee.first_name`
txtLastName	ControlSource	`Employee.last_name`
frmMulti	DataSession	2 - Sessão de Dados Privada

Ao escolher **Executar formulário** no formulário Acionador, será criada uma instância do formulário Multi. Quando o formulário Acionador for fechado, a matriz de propriedades aForms será liberada e todas as instâncias de Multi serão destruídas.

O Visual FoxPro oferece algumas funções e propriedades para ajudá-lo a gerenciar múltiplas instâncias de objetos. Para obter informações, consulte os tópicos AINSTANCE( ), AUSED( ) e DataSessionID na Ajuda.

Definindo a área de criação de formulários

Você pode definir a área máxima de criação para o **Criador de formulários** na caixa de diálogo **Opções**.

Guia Formulários da caixa de diálogo Opções

Para definir a área máxima de criação de um formulário

1 No menu **Ferramentas**, selecione **Opções**.

2 Na caixa de diálogo **Opções**, escolha a guia **Formulários**.

3 Na caixa **Área máxima de criação**, escolha as coordenadas, em pixels, para definir a área máxima de criação.

Quando a área máxima de criação é definida, o segundo plano do **Criador de formulários** fica branco dentro dos limites da área de criação e cinza nas áreas externas a essa área. Se você desenvolver aplicativos utilizando um monitor com resolução de 1024 x 768, por exemplo, poderá definir sua resolução de criação em 640 x 480 e saber que os formulários que você criar sempre caberão em telas 640 x 480..

Dentro da área de criação, não se esqueça de levar em conta os atributos padrão da janela, tais como barras de ferramentas. Por exemplo, em telas com resolução de 640x480, os formulários que tiverem uma barra de status e uma barra de ferramentas ancorada na parte superior ou inferior da tela poderão ter uma altura máxima de 390 pixels.

Atributo da janela principal do Visual FoxPro	Pixels necessários
Título e menu	38
Barra de status	23
Barra de ferramentas fixa	29

Utilizando dados locais e remotos em um formulário

Você pode criar formulários que podem facilmente alternar entre a utilização de dados locais e de dados que são armazenados remotamente (por exemplo, em um servidor de banco de dados). Este procedimento permite que você crie um protótipo de aplicativo utilizando dados locais ou de teste e, em seguida, alterne para dados remotos ou vivos sem fazer alterações substanciais nos formulários.

Por exemplo, se o seu aplicativo do Visual FoxPro for auxiliar para uma tabela de clientes grande em um servidor de banco de dados, você poderá criar um arquivo .DBF local que contenha uma amostra pequena mas representativa dos dados. Em seguida, poderá criar, testar e depurar os formulários com base nesse pequeno conjunto de dados. Quando você estiver pronto para distribuir o aplicativo, poderá vincular o formulário ao conjunto grande de dados.

A chave para poder alternar entre os dados locais e remotos é certificar-se de que você está utilizando visualizações em vez de vincular diretamente o formulário (e seus controles) a uma tabela. Para acessar dados remotos, é necessário que você utilize uma visualização em qualquer evento. Por isso, para facilitar a alternância entre os dados locais e remotos, crie também uma visualização para os dados locais. Ao criar o formulário, você poderá adicionar visualizações ao ambiente de dados, e, em seguida, alternar entre elas, conforme necessário.

Para criar um formulário que possa alternar entre dados locais e remotos

1. Crie duas visualizações dos dados, uma que aponte para os dados remotos e outra que aponte para os dados locais.
2. Crie um novo formulário.
3. Abra o **Criador de ambientes de dados** para o formulário e adicione as visualizações.
4. Clique o botão direito sobre o **Criador de ambientes de dados** e, em seguida, escolha **Propriedades**.
5. Na **janela Propriedades**, defina a propriedade Alias para os dois cursores com o mesmo nome.
6. Defina a propriedade OpenViews do **Ambiente de dados** como **1 – Somente local** ou **2 – Somente remoto**, dependendo da visualização que você desejava utilizar ao executar o formulário.

   Observação Como você está utilizando a mesma alias para as duas visualizações, não será preciso escolher **0 – Local e Remoto** (padrão).
7. No formulário, adicione os controles necessários e defina as propriedades ControlSource para os campos apropriados na visualização. Como ambas as visualizações têm a mesma alias, os controles irão responder automaticamente a qualquer visualização que estiver ativa quando o formulário for executado.

Após a criação do formulário, você poderá alternar entre as alias das visualizações, alterando a propriedade OpenViews do ambiente de dados. É possível realizar esta função no Ambiente de dados ao utilizar o **Criador de formulários**. De forma alternativa, você pode escrever código e anexá-lo a um evento, o que é útil se você desejar alternar entre as visualizações em tempo de execução. Por exemplo, você pode colocar esse código no evento Activate do formulário:

```
THISFORM.DataEnvironment.OpenViews = 2 && Utilize visualização remota
```

Se você criar um formulário que possa alternar entre os dados remotos e locais, deverá também criar o código de navegação para acomodar as visualizações, principalmente se você estiver criando formulários com relacionamento um-para-n. Por exemplo, se o formulário tiver acesso apenas a uma tabela ou visualização local, você poderá utilizar o código, como o que está exibido abaixo, em um botão de comando Próximo para mover-se para o próximo registro em um cursor:

```
SKIP 1
THISFORM.Refresh()
```

No entanto, esse código não será eficaz quando você estiver navegando em uma visualização remota, pois ele presume que cursor contenha todos os dados exigidos pelo formulário. Como regra geral, você deseja minimizar a quantidade de dados que carrega a partir da fonte de dados remota.

A solução é utilizar uma visualização parametrizada. Por exemplo, a definição de uma visualização utilizada para editar informações sobre o cliente poderia ser:

```
SELECT * FROM CUSTOMERS WHERE ;
   CUSTOMERS.COMPANY_NAME = ?pCompanyName
```

Quando o formulário for executado, poderá solicitar que o usuário forneça o nome de um cliente utilizando uma caixa de diálogo ou permitindo que o usuário insira o nome em uma caixa de texto. O

código para um botão Exibir seria semelhante ao seguinte:

```
pCompanyName = THISFORM.txtCompanyName.Value
REQUERY("customer")
THISFORM.Refresh()
```

Para obter maiores informações sobre visualizações parametrizadas, consulte “Criando uma visualização parametrizada” no capítulo 8, Criando visualizações

Definindo modelos de formulários

Você pode criar sua própria classe de formulários, para ser utilizada como modelo para todos os formulários que criar, ou utilizar um dos exemplos de classe que acompanham o Visual FoxPro.

Quando você for criar um novo formulário, o mesmo será baseado no modelo do formulário que tiver sido definido na caixa de diálogo **Opções**. Se não houver um modelo especificado, o novo formulário será baseado na classe principal Form do Visual FoxPro. Para obter maiores informações sobre as classes do Visual FoxPro, consulte o capítulo 3, Programação orientada a objetos.

Vantagens da utilização de modelos do formulário

Os modelos de formulários permitem que você defina as propriedades padrão dos seus formulários, de modo que possa dar facilmente a todos os formulários do seu aplicativo uma aparência e funcionamento uniformes. Você poderia incluir o logotipo da empresa, por exemplo, e utilizar uma combinação de cores uniforme em todos os formulários, desenvolvendo um modelo de classe de formulário com esses atributos. Se o logotipo da empresa for alterado, bastará que você altere a figura na classe de formulário modelo para que todos os formulários criados com base nessa classe sejam atualizados automaticamente com o novo logotipo.

Pode-se adicionar propriedades e métodos personalizados à classe de formulários do Visual FoxPro, de modo que fiquem disponíveis para todos os formulários do seu aplicativo. Se você costuma criar variáveis e procedimentos definidos pelo usuário, que são dimensionados para um formulário, a utilização de propriedades e métodos personalizados permite essa funcionalidade, com a vantagem de permitir a criação de um modelo de encapsulamento mais limpo.

Especificando o modelo de formulário padrão

Você pode especificar uma classe de formulários a partir de uma biblioteca de classes registrada para o seu modelo de formulário.

Para especificar um modelo de formulário padrão

1 No menu **Ferramentas**, selecione **Opções**.

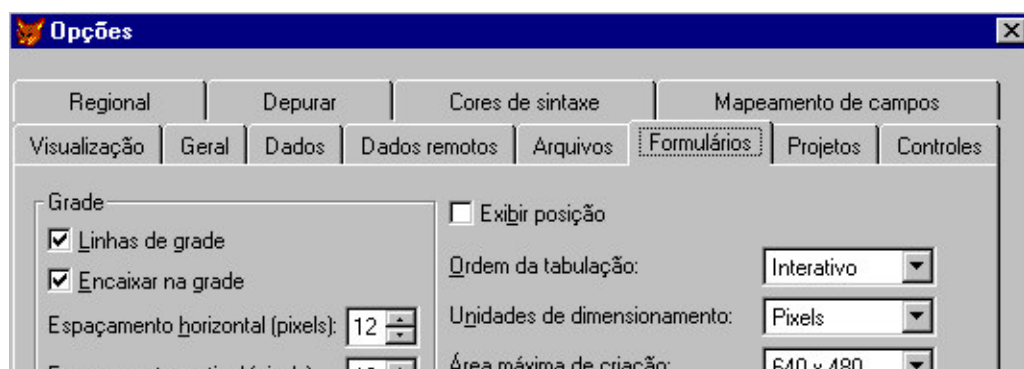
2 Na caixa de diálogo **Opções**, selecione a guia **Formulários**.

3 Na área **Classes-modelo**, escolha a caixa de verificação **Formulário**.

Caso nenhum modelo de formulário tenha sido selecionado, a caixa de diálogo **Abrir** será aberta para que você possa escolher uma classe de formulário. Caso um modelo de formulário tenha sido selecionado, você poderá alterá-lo selecionando o botão de reticências e escolhendo uma outra classe.

4 Escolha **Definir como padrão** caso deseje que o modelo seja utilizado em sessões subseqüentes do Visual FoxPro.

5 Escolha **OK**.

Guia Formulários da caixa de diálogo Opções

Utilizando modelos de formulários

Você pode especificar modelos de conjunto de formulários da mesma forma que especifica modelos de formulários. São possíveis as combinações a seguir:

- Especificar modelos para formulários e conjuntos de formulários.

  A opção **Formulário** da caixa de diálogo **Novo** (e qualquer outro recurso para se criar um novo formulário) criará automaticamente um conjunto de formulários baseado no modelo de classe de conjuntos de formulários. Quando você escolher **Adicionar novo formulário** no menu **Formulário** do **Criador de formulários**, será adicionado ao conjunto de formulários um formulário baseado no seu modelo.

- Especificar somente o modelo de conjunto de formulários.

  A opção **Formulário** da caixa de diálogo **Novo** (e qualquer outro recurso para se criar um novo formulário) criará automaticamente um conjunto de formulários baseado na classe modelo de conjuntos de formulários. Quando você escolher **Adicionar novo formulário** no menu **Formulário** do **Criador de formulários**, será adicionado ao conjunto de formulários um formulário baseado na classe principal de formulários do Visual FoxPro.

- Especificar somente o modelo de formulário.

  A opção **Formulário** da caixa de diálogo **Novo** (e qualquer outro recurso para se criar um novo formulário) criará automaticamente um formulário baseado na classe modelo de formulários.

- Não especificar qualquer modelo.

  A opção **Formulário** da caixa de diálogo **Novo** (e qualquer outro recurso para se criar um novo formulário) criará automaticamente um formulário baseado na classe principal de formulários do Visual FoxPro.